

624.04
Д53

Дмитриев Л. Г.

Прозрачливание расчета пространственных конструкций



ГОССТРОЙИЗДАТ УССР • КИЕВ 1963

624.01

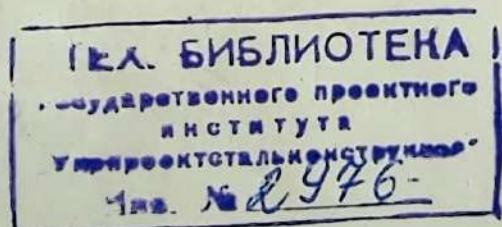
Зональный научно-исследовательский и проектный институт
типового и экспериментального проектирования жилых
и общественных зданий Госкомитета по гражданскому
строительству и архитектуре при Госстрое СССР

Л. Г. ДМИТРИЕВ, П. М. СОСИС

624.04

Д 53

ПРОГРАММИРОВАНИЕ РАСЧЕТА ПРОСТРАНСТВЕННЫХ КОНСТРУКЦИЙ



Государственное издательство
ЛИТЕРАТУРЫ ПО СТРОИТЕЛЬСТВУ И АРХИТЕКТУРЕ УССР
Киев — 1963

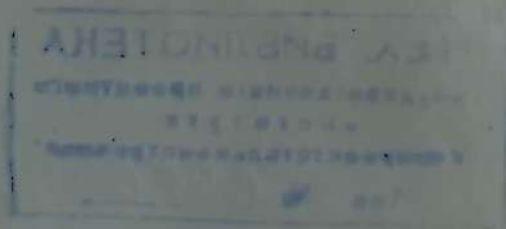
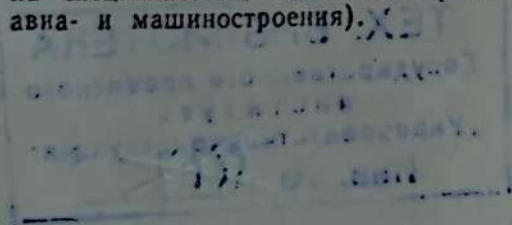
В книге изложены численные методы расчета сооружений с помощью современных цифровых машин (электрических и электронных).

Большое внимание уделено анализу алгоритмических языков, вопросам создания машинных алгоритмов и технологии программирования. Предложены приемы экономной организации памяти электронных вычислительных машин, основанные на изучении особенностей структур матриц, характерных для задач строительной механики и прикладной теории упругости.

На алгоритмическом языке АЛГОЛ-60 разработан ряд программ решения систем линейных уравнений определенных структур и обобщенные программы расчета пологих пространственных систем в нелинейной постановке.

Книга предназначена для инженеров-проектировщиков, работников научно-исследовательских институтов и студентов строительного профиля обучения.

Общая часть книги, относящаяся к вопросам программирования, решения систем линейных уравнений строительной механики и т. п., может быть использована специалистами смежных отраслей техники (судо-, авиа- и машиностроения).



ПРЕДИСЛОВИЕ

Внедрение в практику строительства эффективных пространственных конструкций вызывает необходимость в более совершенных методах расчета с применением математических машин.

С появлением быстродействующих электронных вычислительных машин (ЭВМ) на первый план выдвигаются численные методы (разностные и вариационные), с помощью которых можно решать любые краевые задачи теории упругости.

При использовании определенных численных методов (например, метода сеток [3] или ему родственных) можно создать универсальные алгоритмы, каждый из которых пригоден для решения обширного круга задач.

Первостепенная важность численных методов общепризнана. Более того, в каждой прикладной отрасли знания (в частности в строительной механике) возникает и развивается новое направление, связанное с разработкой машинных алгоритмов, с использованием алгоритмических языков и программирования.

Настоящая работа в значительной мере посвящена этому новому направлению, в ней много внимания уделено анализу алгоритмических языков и прежде всего международного языка АЛГОЛ-60 применительно к задачам строительного проектирования.

Многими странами, в том числе Советским Союзом, принято соглашение о создании программирующих программ на этом языке для всех существующих и вновь выпускаемых машин, которые должны быть снабжены устройством буквенного ввода данных.

Распространение и популяризация языка АЛГОЛ-60 среди широких кругов инженеров-строителей будет содействовать внедрению ЭВМ в практику строительного проектирования, что очень важно, поскольку оснащение народного хозяйства СССР электронными вычислительными машинами происходит весьма быстрыми темпами.

Другой важный вопрос, изложенный в работе, относится к созданию машинных алгоритмов, то есть таких, которые позволяют наиболее рационально использовать возможности вычислительных машин.

При составлении машинных алгоритмов и основанных на них программ следует учитывать, что современные ЭВМ имеют еще ряд ограничений: объем оперативной памяти машины нередко недостаточен для размещения больших по объему задач, подготовка задачи к решению (перфорирование) занимает, подготавливая, в некоторых машинах запятая фиксирована, много работает печатающее устройство и т. д.

В связи с этим большое значение имеют входящие в состав машинного алгоритма специальные приемы экономной организации памяти, уменьшения исходной информации, печати и т. д.

В данной книге исследуется класс пологих пространственных конструкций, расчет которых представляет весьма большие трудности и где применение вычислительных машин особенно эффективно. Рассматриваются разнообразные машинные задачи: расчет вантовых систем, ребристых оболочек, балок-стенок и сложных стержневых систем.

Обычно при расчете сборных железобетонных оболочек, имеющих ребра, задача сначала приводится к континуальной (ребра довольно произвольно «размазываются»), после чего нередко решается методом сеток, т. е. фактически как дискретная.

Изложенный в настоящей работе метод в известной степени устраняет это противоречие. Хотя для решения применяется операторная форма записи, характерная для методов теории упругости, задача с самого начала рассматривается как дискретная.

Это позволяет избежать многих трудностей при формулировке граничных условий и дает возможность описать одной системой уравнений достаточно широкий класс задач.

В книге программы изложены на языке АЛГОЛ-60. Рабочие программы и примеры расчета находятся в зональном научно-исследовательском и проектном институте типового и экспериментального проектирования жилых и гражданских зданий (Киев, Б. Житомирская, 9).

Главы I—III, IX и X работы написаны П. М. Сосисом, главы IV—VIII—Л. Г. Дмитриевым.

Авторы выражают глубокую благодарность докт. техн. наук проф. П. М. Варваку и канд. физ.-мат. наук А. А. Летичевскому за ценные указания, сделанные при рецензировании рукописи.

Глава I

О НОВОМ ПОДХОДЕ К МЕТОДАМ РАСЧЕТА В СВЯЗИ С РАЗВИТИЕМ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И ПРОГРАММИРОВАНИЯ

Еще не так давно расчеты статически неопределимых стержневых систем выполнялись в проектных институтах инженерами сравнительно высокой квалификации. Основным средством для выполнения расчетов служила логарифмическая линейка. Системы уравнений в обычном виде не составлялись, а коэффициенты системы (погонные жесткости) записывались на схеме рамы. Поскольку расчеты выполнялись инженерами, к математическому алгоритму прибавлялась так называемая инженерная трактовка, описывающая, например, процесс уравновешивания, восстановления либо устранения связей основной системы и т. п. Положение изменилось с распространением электрических вычислительных машин с ручным набором и съемом данных (например, клавишные машины ВММ-2, фирмы Рейнметалл-Суперметалл марок САСЛ, САР, КЕЛ). На базе этих машин во многих проектных институтах созданы отделы механизации расчетов. По-видимому, на ближайший период (5—10 лет) указанные машины будут являться основным средством механизации инженерно-технических расчетов.

На клавишных автоматах и полуавтоматах расчеты выполняют не инженеры, а операторы вычислительных машин. Основная трудность, таким образом, состоит в передаче информации от инженера к оператору. Преодолеть эту трудность можно двумя способами:

- 1) объединить в одной группе инженеров и операторов;
- 2) использовать особый формализованный язык, понятный оператору.

При первом способе наиболее простые вычисления укладываются в простейшие табличные бланки, в заглавных строках которых указана последовательность вычислений.

В более сложных случаях, не укладывающихся в табличные формы, инженер обучает оператора той или иной методике и наблюдает за его работой. На стыке между различными эта-

пами расчетов инженер обрабатывает полученные от оператора результаты и выдает новое задание и т. п.

Недостаток указанной организации вычислительного процесса состоит в необходимости обучения операторов инженерным методам расчета. Такое обучение не всегда возможно из-за неподготовленности оператора, кроме того, оно предполагает создание жесткой технологии, основанной на определенном методе расчета. Недостатком также является необходимость постоянного наблюдения и контроля над оператором со стороны инженера.

Второй способ не имеет этих недостатков. Пользуясь формализованным языком, можно описать любой алгоритм и представить его в виде стандартной программы вычислений. Такой язык был предложен нами в 1959—1961 гг. [59, 60, 61] и нашел применение во многих проектных институтах.

В основу этого языка положен принцип адресности, широко применяемый при программировании расчетов для электронных цифровых машин. Однако при использовании клавишных автоматов рассматривается не линейная последовательность ячеек, а таблица (матрица) и ее элементы — клетки, строки, графы.

Простейшая программа записывается в виде равенства, в левой части которого указывается, откуда взять числа (их адреса) и какие действия над ними произвести, в правой — куда записать результаты вычислений (адреса результатов). Программа вычислений, описывающая тот или иной алгоритм, состоит из набора простейших программ, в число которых входят и проверочные равенства. Предложенная символика исходит от матричного исчисления и имеет много общего с ним, но оперирует терминами более привычными для инженера и оператора-вычислителя, а благодаря адресной форме записи позволяет оперировать с любыми группами чисел в разных таблицах, не прибегая в ряде случаев к формальным преобразованиям матриц.

Эффективная программа расчетов сооружений должна отвечать следующим требованиям:

а) исчерпывающе и однозначно описывать алгоритм, быть простой и доступной для оператора. В ряде случаев для простоты программы можно немного увеличить объем вычислений;

б) обладать свойством цикличности, повторяемости. Следует стремиться к тому, чтобы вычисления выполнялись по небольшому числу формул с изменением входящих в них адресов;

в) включать построчные (пографные) проверки. Если проверочная формула не соблюдается, то оператор должен повторить вычисления для отрезка программы, расположенного между двумя проверочными формулами, и перейти к следующему этапу лишь после того, как последняя проверка будет удовлетворена. Такое построение программы обеспечивает значительно лучший контроль, чем так называемая проверка «второй ру-

кой», при которой возможно повторение тех же или даже появление новых ошибок. Включение независимых самопроверок в программу практически исключает возможность ошибок в конечных результатах, отпадает также необходимость в дублировании расчетов;

г) использовать возможности клавишных автоматов (полуавтоматов) по автоматическому накоплению сумм произведений чисел (сумм частных), что позволяет уменьшить количество записей и вводов данных. При составлении программы предпочтение должно быть отдано так называемым компактным схемам алгоритмов;

д) быть типовой, стандартной, то есть описывать расчет не одной какой-либо задачи, а целого класса задач. Это дает возможность использовать программу многократно для любых задач данного класса. При наличии стандартной программы подготовка конкретной задачи для решения сводится к привязке табличных бланков, в которых нумерация строк и граф должна соответствовать нумерации элементов или усилий системы.

Указанные требования не удовлетворяются, если использовать для составления программы методы, приспособленные для логарифмической линейки, например, методы Кросса, Кани и др. В этих методах отсутствует компактность: каждое вычисление производится отдельно и результат его записывается в таблицу. Это значит, что производится огромное количество ненужных записей, а при пользовании клавишными машинами записи результатов занимают значительно больше времени, чем сам счет (выполняемый автоматически).

Для машин наиболее целесообразны те методы, в которых математическая суть четко выражена и состоит в преобразовании матриц (таблиц), а именно — методы строительной механики в классической форме, которые по существу совпадают с методами линейной алгебры. В СССР вопросу использования теории матриц для решения задач строительной механики посвящены работы советских ученых А. Ф. Смирнова [55], А. П. Филина [54], Л. К. Нарца [42], Ш. М. Гофмана [12], Е. С. Гребеня [13], В. Д. Шайкевича [73] и др. Английским ученым Дж. Аргиромом и его последователями [54] предложена матричная теория статик сооружений, излагающая по существу методы строительной механики в терминах линейной алгебры и матричного исчисления. Такая постановка вопроса представляет бесспорный интерес для исследования методов строительной механики. Представление же всех этапов расчета статически неопределимых систем в виде операций над таблицами (матрицами) вполне может быть выполнено в границах классических методов строительной механики [61] и для практических целей не обязательно переосмысливать их в духе матричной теории статик сооружений.

Благодаря стандартным программам клавишные автоматы превращаются как бы в машины с программным управлением, то есть выполняют те же функции, что и электронные машины, с тем отличием, что только счет выполняется автоматически (причем отсутствует быстроедействие), а набор данных на клавиатуре и запись результатов арифметических операций выполняются оператором вручную.

Несмотря на относительно (по сравнению с электронными машинами) медленное выполнение расчетов, клавишные автоматы сохраняют свое значение, по-видимому, в течение длительного периода, главным образом, потому, что решение задач на электронных машинах пока еще требует дополнительной многоступенчатой работы (кодировка исходных данных, перфорирувание, печать и т. п.), занимающей много времени. Поэтому для задач, решение которых на клавишных автоматах требует сравнительно небольшого времени (от нескольких часов до нескольких дней), особенно в тех случаях, когда объем ввода и вывода велик, может оказаться нецелесообразным использование электронной вычислительной машины. Впрочем, и этот подход может измениться, поскольку в настоящее время работают над созданием устройства, обеспечивающего непосредственное чтение буквенного и цифрового материала и ввода его в машину [26].

Электронные вычислительные машины позволяют не только резко сократить время на выполнение сложных расчетов, но и решать задачи, практически не выполнимые на клавишных автоматах. В частности, оказывается возможным быстро решать сложные задачи прикладной теории упругости, которые приводятся к системам линейных уравнений весьма высоких порядков, тригонометрическим рядам и т. д. Вычисление коэффициентов и решение систем линейных уравнений с многими десятками или сотнями неизвестных на клавишных машинах практически невозможно, так как требует многих месяцев или даже лет напряженного труда.

Пока не было электронных машин, приходилось либо ограничиваться примитивными методами расчета, которым инженер не вполне доверял и поэтому допускал излишние запасы прочности, либо отказываться (что нередко и имело место) от применения новых конструкций в пользу менее рациональных, но изученных решений.

Программирование для электронных вычислительных машин и методы решения задач на этих машинах имеют определенную специфику.

Прежде всего необходимо иметь в виду, что современные машины пока еще имеют ряд ограничений.

Ограничен объем оперативной памяти машины, в которую засылаются исходные данные, промежуточные и конечные ре-

зультаты. У большинства машин память состоит из 1024 и 2048 ячеек («Урал», «Минск», «Киев», «Стрела», БЭСМ-2), у более совершенных машин — из 4096 ячеек. Правда, имеется еще внешняя память (магнитные барабаны, ленты), из которой материал можно последовательно засылать в оперативную память. Для этого необходимо разделить задачу на независимые части, что нередко приводит к громоздкой схеме счета. Кроме того, многократное обращение к внешней памяти сильно замедляет решение; поэтому оперативная память машины должна использоваться весьма экономно с последовательной засылкой промежуточных и конечных результатов по возможности в одни и те же ячейки с вытеснением ранее расположенных в них чисел, если они для дальнейшего расчета не нужны. Машина устроена так, что при засылке нового числа в ячейку памяти ранее хранившееся в ней число автоматически стирается.

В ЭВМ весь процесс счета автоматизирован. Однако скорость действия отдельных частей машины не одинакова. Наибольшей быстротой действия отличается сам счет. В современных машинах он выполняется со скоростью многих тысяч операций в секунду (БЭСМ-2 в среднем 8000, «Киев» — 10000, имеется также машина со скоростью 20000 вычислений в секунду). Наряду с этим, ввод данных и печатание результатов выполняется относительно медленно — примерно 15—20 кодов (цифр) в секунду.

Отсюда следует, что количество вычислений не всегда является основным критерием, определяющим выбор метода решения задачи. Нередко на первый план выдвигается требование сокращения объема вводимой информации (программы, исходных данных) и печатания результатов. В самом деле, при достаточно большом числовом материале (что характерно для многих сложных задач строительной механики и прикладной теории упругости) ввод данных и печатание результатов могут занять во много раз больше времени, чем сам счет. Поэтому может оказаться целесообразным в ряде случаев значительно увеличить количество вычислений (даже в 2—3 раза), чтобы уменьшить объем вводимой информации. Как видно, подход принципиально отличается от того, какой применяется при решении инженерных задач на клавишных машинах.

Подготовка исходных данных — перфори́рование — выполняется со скоростью набора данных на клавишном автомате (даже несколько медленнее). Хотя перфоратор отделен от машины и время пробивки перфокарт учитывается отдельно и не включается во время работы машины, но общее время, необходимое для решения задачи, все же увеличивается. Нет сомнения, что указанные противоречия существующих ЭВМ будут со временем устраняться, но пока приходится учитывать их при составлении программ.

В настоящей работе предложен ряд приемов экономной организации памяти. Так, при составлении программ решения систем линейных уравнений используются определенные свойства матриц, характерные для задач строительной механики.

Другие приемы (так называемый операторный метод записи) применяются для того, чтобы уменьшить объем вводимых чисел, — вместо матрицы вводится информация о ней, а затем уже машиной по специальной программе матрица формируется.

Указанные приемы применимы не только при решении систем линейных уравнений, но и при других преобразованиях матриц.

Таким образом эффективная программа для ЭВМ должна отвечать следующим требованиям:

а) выражать алгоритм задачи с помощью минимального количества команд и, следовательно, занимать сравнительно небольшой объем оперативной памяти машины;

б) обладать свойством цикличности. Это требование для ЭВМ имеет еще большее значение, чем для электрических автоматов, так как от него зависит выполнение пункта «а»;

в) обеспечивать экономное размещение числового материала в оперативной памяти путем замещения одних чисел другими и исключения нулевых элементов матриц;

г) предусматривать минимальный ввод чисел, заменяя (в тех случаях, когда это возможно) ввод матрицы вводом краткой, но исчерпывающей информации о ней;

д) быть стандартной, то есть описывать решение не одной какой-либо задачи, а целого класса задач. Нередко бывает так, что добавлением небольшого количества команд мы делаем программу более общей. В этом случае целесообразно идти на некоторое (небольшое) увеличение программы, разветвляя процесс для двух-трех обширных групп;

е) обеспечивать проверку ввода и суммирования с помощью контрольных сумм либо повторного счета и сравнения результатов. Вследствие быстроты счета проверки не следует дробить на построчные (как это делается в электрических автоматах), а охватить ими сразу большие массивы чисел.

Требование компактности схем счета, предъявляемое при выполнении расчетов на электрических автоматах, для ЭВМ не является необходимым. Разница между компактной и обычной схемами счета (например, компактной и обычной схемами Гаусса для решения систем уравнений) будет состоять в применении к ЭВМ лишь в характере записи чисел в ячейках памяти: в первом случае суммы произведений будут накапливаться в дополнительных рабочих ячейках, во втором — во всех ячейках матрицы. Количество необходимых ячеек при обычной схеме счета не увеличивается (даже несколько уменьшается благодаря исключению дополнительных рабочих ячеек), а алгоритм решения не-

сколько упрощается. Поэтому при использовании ЭВМ компактные схемы обычно не нужны.

Для того чтобы удовлетворить изложенным выше требованиям, программист должен хорошо изучить поставленную перед ним задачу и проявить определенную изобретательность; в противном случае даже грамотно составленная программа может оказаться нежизненной — большие по объему задачи (именно в этих случаях имеет смысл применять ЭВМ) нельзя будет разместить в памяти машины. Для того чтобы воспользоваться такой программой, придется разбивать большую задачу на части и получаемые результаты вновь вводить в машину. Конечно, можно предусмотреть привлечение внешней памяти, но это значительно усложняет программу и замедляет счет.

В работе П. М. Ханта [54] приведена матричная интерпретирующая программа, составленная на основе метода Дж. Аргираса. Пользуясь этой программой, на ЭВМ «Pegasus» были выполнены расчеты для сложных авиационных и судостроительных конструкций, что весьма интересно с инженерной точки зрения. Однако, как указано самим автором, расчет сложных конструкций занимает много времени. Так, для расчета треугольного крыла без ослаблений потребовалось 140 часов машинного времени, причем «... большую часть времени машина затрачивает на перфорирование промежуточных результатов на бумажной ленте и считывание этих данных, когда это необходимо» [54, стр. 364], то есть имеет место тот недостаток, о котором говорилось выше*. По-видимому, это объясняется тем, что матричная интерпретирующая программа предполагает полное заполнение матриц (за исключением диагональных). Для размещения матрицы с размерами m и n в памяти требуется $m \times n$ ячеек даже в тех случаях, когда имеется много нулевых элементов, что характерно для больших задач строительной механики. Матричные инструкции, входящие в интерпретирующую программу, описывают известные матричные операции без каких-либо изменений, вследствие чего элементы, постоянно остающиеся нулевыми, загромождают ячейки оперативной памяти.

Если даже согласиться с тем, что матричный язык является идеальным для электронных вычислительных машин (вообще говоря, это не бесспорно, поскольку память ЭВМ не матричная, а линейная), то это вовсе не означает, что следует ограничиться существующими матричными операциями. Нередко изменение операции или введение новой операции может сильно упростить решение задач на ЭВМ.

* Как справедливо замечено в рецензии Р. А. Резникова (см. «Строительная механика и расчет сооружений» 1963 г., № 1), при других алгоритмах решение той же задачи потребовало бы примерно два часа машинного времени.

Как известно, библиотека стандартных программ, а также компилирующие и интерпретирующие системы (применяемые для размещения всей библиотеки или ее части в оперативном запоминающем устройстве) существуют во многих машинах, в частности в серийной отечественной машине БЭСМ-2, но в их состав входят программы, описывающие не только матричные, но и многие другие операции.

Помимо перечисленных выше общих требований, рабочие программы должны учитывать конструктивные особенности той машины, для которой они составлены, — адресность (одно-, двух- или трехадресные), режим работы машины (с фиксированной или плавающей запятой). При программировании для машин с фиксированной запятой (например, для ЭВМ «Киев», «Урал» и т. п.) следует учитывать, что все числа — исходные, промежуточные и результаты — должны быть меньше единицы, для чего для чисел выбирается определенный масштаб. Выбор масштаба не для всех алгоритмов является простым и легким делом. Поэтому программирование и подготовка задач для машин с плавающей запятой, как правило, выполняются проще, чем для машин с фиксированной запятой, а диапазон алгоритмов шире.

Программы для ЭВМ отличаются от соответствующих программ для электрических автоматов прежде всего логической схемой. При составлении программы для человека (оператора электрического автомата) многое является очевидным, например, последовательность чисел, следующих через определенные интервалы. Для электронной же вычислительной машины нет ничего «очевидного», она анализирует заново каждую новую ситуацию, как бы распознает путем логических сопоставлений по принципу «да—нет» закономерность алгоритма (в частности, закономерность последовательности чисел).

Управление работой машины осуществляется с помощью проверок выполнения тех или иных логических условий. Если условие выполняется, управление передается одной команде, если нет — другой. Величины, входящие в проверочные условия (команды управления) подсчитываются машиной так же, как и величины, входящие в формулы арифметического счета. Обычно логических действий в программах для ЭВМ значительно больше, чем чисто арифметических. В этом состоит своеобразие программ для электронных вычислительных машин.

Логическая схема счета определяется принятым алгоритмом решения задачи и не зависит от типа машины; ее можно составить по определенным правилам, даже не будучи знакомым с кодами и другими данными для ЭВМ, надо лишь хорошо знать сущность решаемой задачи.

Поэтому имеется стремление отделить эту наиболее творческую часть работы от собственно рабочего программирования и

представить в виде обобщенной программы, составление которой станет доступным для широкого круга специалистов различных областей знания. Так возникли различные блок- и граф-схемы, операторное программирование, в котором отдельные части программы представляются в виде символов с логическими связями между ними, адресное программирование, международный алгоритмический язык АЛГОЛ-60.

Система адресного программирования, разработанная в Институте кибернетики АН УССР [8] и пользующаяся заслуженной популярностью среди математиков и инженеров, позволяет составлять программы в виде, пригодном для любых электронных вычислительных машин. Адресная запись, абстрагируясь от конкретной машины, дает не только общие контуры программы (как например, блок-схема), но и полностью разработанную логическую структуру программы. Она содержит исчерпывающие сведения о программе и может быть путем формальных операций, подчиняющихся определенному алгоритму, переведена на язык той или иной машины, то есть записана в ее кодах. Это позволило создать в Институте кибернетики АН УССР программирующие программы для ЭВМ. Получив информацию в виде адресной программы, машина сама преобразовывает ее в рабочую и выдает колоду перфокарт с пробитой программой.

Машины, для которых разработаны указанные программирующие программы, не имеют еще устройства буквенного ввода. Поэтому адресную программу перед вводом в машину придется кодировать и представлять в виде набора цифр, но кодирование превращается в механический процесс и может выполняться не математиком-программистом, а оператором-кодировщиком. Объясняется это тем, что при кодировании для программирующей программы исключается наиболее сложный этап рабочего программирования, состоящий в распределении адресов программы и числового материала в памяти машины, расшифровке сложных адресных формул и т. п.

Все же при отсутствии устройства буквенного ввода программирование автоматизировано только частично.

Наиболее полное выражение идея автоматизации программирования нашла в международном алгоритмическом языке АЛГОЛ-60. Советским Союзом и рядом стран Европы и Америки принято соглашение о создании программирующих программ на этом языке для всех существующих и вновь выпускаемых машин, которые снабжены специальным устройством буквенного ввода данных.

Составление программирующих программ является сложным делом и требует определенного времени. Пока же целесообразно использовать алгол для изложения обобщенных программ с тем, чтобы формальный перевод их на язык конкретной ЭВМ выполнялся программистом.

Примечательным является то, что алгол является алгоритмическим (формульным) и в известном смысле словесным языком, полностью абстрагированным от машины; степень отвлечения его значительно больше, чем адресного языка.

По мере распространения международного языка запись алгоритмов на нем станет столь же доступной и привычной для математиков и инженеров, как ныне запись с помощью алгебраических формул со словесными связями между ними. Программирование задач перестанет быть функцией узких специалистов, широкая публикация и обмен программами между научными коллективами разных стран будет способствовать созданию обширных библиотек стандартных программ для различных областей науки и техники, что необходимо для эффективного использования электронных вычислительных машин.

Характерно, что алгол по существу охватывает другие известные алгоритмические языки. Так, адресный язык является частным случаем алгола. Инженерное программирование имеет много общего с алголом (описание двумерных массивов и их элементов) и может рассматриваться как одно из его конкретных представлений, предназначенное для малой механизации — электрических автоматов.

Однако АЛГОЛ-60 значительно шире других алгоритмических языков по своим возможностям и в этом его преимущество. Развитие и совершенствование этого языка продолжается [20].

В НИИ экспериментального проектирования Б. Академии строительства и архитектуры УССР сложилась и успешно применяется методика трехстадийного программирования по аналогии с известными тремя стадиями проектирования

Первая стадия — арифметизация задачи — выполняется в форме эскизной программы. Это по существу программа для электрической цифровой машины (так называемая «инженерная программа»), но записанная в более краткой форме с указанием закономерности образования адресов. Такая форма записи дает возможность проверить принципиальную схему программы первоначально на электрическом автомате. Моделью памяти в эскизной программе служит матрица и ее элементы — строки, графы, клетки. При составлении программы в целом решается вопрос об экономии ячеек памяти путем замещения одних чисел другими в одних и тех же ячейках (клетках). При составлении эскизных программ в большинстве случаев можно отказаться от компактных схем.

Проводя аналогию с проектированием, можно сказать, что эскизная программа представляет собой как бы стадию проектного задания.

Вторая стадия состоит в разработке обобщенной программы. Применяется краткая или (при отсутствии программирующей программы) развернутая запись на алголе. Последняя совпа-

дает с адресной программой. Во второй стадии, соответствующей техническому проекту, разрабатывается логическая структура программы, решаются вопросы организации памяти машины для экономного ее использования.

Первая и вторая стадия схватывают творческую часть задачи и должны выполняться специалистом (инженером или математиком), не только хорошо знающим алгоритм задачи, но и имеющим опыт численного решения ее на электрических автоматах. Необходимо знать, какой максимальный объем задачи может встретиться на практике, какая требуется точность вычислений и т. п., с тем, чтобы разработать наиболее эффективный способ решения на ЭВМ.

Последняя стадия — составление рабочей программы — соответствует стадии рабочего проекта. При наличии проверенной программирующей программы эта стадия выполняется самой машиной. При отсутствии программирующей программы запись обобщенной программы в кодах данной ЭВМ (кодировка) выполняется программистом.

Применение быстродействующих электронных машин приводит к изменению подхода к методам расчета, их переоценке. Объем вычислений уже не является главным критерием, на первый план выдвигаются такие вопросы, как упрощение алгоритма и основанной на нем программы, уменьшение объема информации, экономии памяти и др. Поэтому теряют смысл многие приемы строительной механики, усложняющие алгоритм с единственной целью — уменьшить количество вычислений.

Как видим, выбор расчетного алгоритма в весьма большой степени зависит от применяемых вычислительных средств: один подход целесообразен при использовании логарифмической линейки, другой — электрических автоматов, третий — электронных цифровых машин.

В данной книге не рассматривается применение электронных машин непрерывного действия. Укажем только, что с появлением теории квазианалогов [44] электро моделирующие машины получили мощный толчок для развития: более глубокое толкование получает идея основных (эквивалентных) статически неопределимых систем — прямых электрических аналогов любой мерности, к которым придается дополнительное уравнивающее устройство.

Материал в последующих главах книги излагается под углом зрения положений настоящей главы.

Глава II

ОБОЗНАЧЕНИЯ (ЯЗЫКИ) В СИСТЕМАХ ПРОГРАММИРОВАНИЯ

ИНЖЕНЕРНОЕ ПРОГРАММИРОВАНИЕ

Обозначения элементов таблиц

В общем случае расчет сооружения выполняется во многих таблицах. Поэтому обозначения элементов таблиц должны быть такими, чтобы с их помощью можно было точно указать местоположение (адрес) любого числа или ряда чисел в разных табличных бланках.

В соответствии с этим примем следующие обозначения. Рассчетную таблицу обозначим буквой T_N (индекс N указывает номер таблицы, $N=1, 2, \dots, A, B, \dots$).

Например, T_1 —табл. 1, T_A —табл. A^* .

Бланк таблицы состоит из горизонтальных строк, вертикальных граф и клеток. Под наименованием «клетка» понимается записанное в ней число, под наименованием «графа» и «строка» — множество чисел, записанных в этих элементах.

Строки и графы обозначаются следующим образом:

$$| \text{стр } \alpha \ T_N |_{n_1}^{+\nu}$$

— строка α табл. N в пределах от графы n_1 до графы $n_1 + \nu$ (для сокращения записи индекс n_1 в верхнем пределе не повторяется).

Количество клеток в данном отрезке строки равно $\nu + 1$.

Например, если $\alpha=2$, $N=1$, $n_1=1$, $\nu=3$, то получим

$$| \text{стр } 2 \ T_1 |_1^4$$

Для рис. 1 это числа 5,0; 10,0; —5,0; 8,0.

$$| \text{гр } \alpha \ T_N |_{m_1}^{+\nu}$$

— графа α табл. N в пределах от строки m_1 до строки $m_1 + \nu$.

Например, если $\alpha=3$, $N=1$, $m_1=1$, $\nu=3$, то получим $| \text{гр } 3 \ T_1 |_1^4$.

Для рис. 1 это числа 3,0; —5,0; 15,0; —6,0.

* В тех случаях, когда реальные таблицы имеют буквенное наименование, символ T можно опустить и писать A, B вместо T_A, T_B .

Когда говорится о всей строке таблицы, то обозначения пределов можно опустить. Так, для рис. 1 обозначение стр 2 T_1 (без пределов) охватывает числа 5,0; 10,0; —5,0; 8,0; 18,0.

	T_1						T_2	
	1	2	3	4	5		1	2
1	20,0	-5,0	3,0	-2,0	15,0			
2	5,0	10,0	-5,0	8,0	18,0			
3	4,0	-6,0	15,0	-4,0	9,0		19,0	-10,0
4	1,0	5,0	-6,0	20,0	20,0		-5,0	25,0
5	30,0	4,0	7,0	22,0				

Рис. 1.

Обозначения таблиц можно ставить как внутри прямых скобок, так и вне их. Например, можно написать [стр 2] T_1 ; гр3 T_1 .

В некоторых задачах удобно выделить часть строки, расположенную слева или справа от главной диагонали таблицы.

В этих случаях у номера строки ставится индекс л или п. Например, стр 3_л T_2 — часть строки 3, расположенная слева от главной диагонали T_2 (для рис. 2 это числа 2,0 и 3,0); стр 3_п T_2 — часть строки 3, расположенная справа от главной диагонали (для рис. 2 это числа 3,0; 2,0 и 1,0).

Введем также понятие диагональной графы

$$| \text{гр}DT_N |_{m_1}^{+v}$$

	1	2	3	4	5	6
1	20,0	3,0	2,0	1,0		
2	3,0	10,0	3,0	2,0	1,0	
3	2,0	3,0	20,0	3,0	2,0	1,0
4	1,0	2,0	3,0	15,0	3,0	2,0
5		1,0	2,0	3,0	20,0	3,0
6			1,0	2,0	3,0	15,0

Рис. 2.

— группа клеток, расположенная на главной диагонали табл. N в пределах от строки m_1 до строки $m_1 + v$.



Например, если $N=2$, $m_1=1$ и $v=4$, то получим $|\text{гр}DT_2|_1^5$.

Для рис. 2 это числа 20,0; 10,0; 20,0; 15,0; 20,0.

Клетка таблицы находится на пересечении строки и графы и обозначается двумя буквами (цифрами), разделенными точкой; первая буква (цифра) указывает номер строки, вторая — номер графы.

$i.kT_N$ — клетка на пересечении строки i и графы k табл. N ;

$k.iT_N$ — клетка на пересечении строки k и графы i табл. N .

Например, для рис. 1 в клетке 2.3 T_1 находится число —5,0; в клетке 3.2 T_1 — число —6,0.

Для выделения части таблицы применяется следующее обозначение: $|T_N|_{m_1+v}^{n_1+v}$

— часть таблицы в пределах от строки m_1 до строки $m_1+\mu$ и от графы n_1 до графы n_1+v .

Например, если $N=4$; $m_1=2$; $\mu=3$; $n_1=2$; $v=4$, то получим $|T_4|_{2.5}^{2.6}$.

Для рис. 9 эта часть табл. A , ограниченная жирными линиями, от строки 2 до строки 5 и от графы 2 до графы 6.

С помощью рассмотренных обозначений можно описать разнообразные арифметические и логические операции над числами.

Запись инженерных программ для клавишных автоматов

Простейшие программы охватывают арифметические действия: алгебраическое суммирование, умножение, деление, суммирование произведений или частных, а также некоторые логические операции — отыскание максимального (минимального) числа в группе чисел, выборочное суммирование положительных (отрицательных) чисел или им соответствующих.

Программа вычислений записывается в виде формулы, в левой части которой указывается, откуда взять числа (из каких таблиц, строк, граф, клеток) и какие арифметические действия с ними произвести; в правой части — куда записывать получаемые результаты (адреса результатов).

Алгебраическое суммирование

$\Sigma|\text{гр}2T_1|_1^4 \rightarrow 5.2T_1$ — просуммировать числа графы 2 табл. 1 в заданных пределах и результат записать в клетку 5.2 той же таблицы.

Например, для рис. 1: $-5,0 + 10,0 - 6,0 + 5,0 = 4,0$.

$\Sigma|\text{стр}3T_1|_1^4 \rightarrow 3.5T_1$ — просуммировать числа строки 3 табл. 1 в заданных пределах и результат записать в клетку 3.5 той же таблицы.

Например, для рис. 1: $4,0 - 6,0 + 15,0 - 4,0 = 9,0$.

Выборочное суммирование

$\sum_{\text{по}(+)} |\text{стр} 3 T_1| \rightarrow 3.1 T_3$ — просуммировать положительные числа строки 3 табл. 1 и результат записать в клетку 3.1 табл. 3.

Например, для рис. 1: $4,0 + 15,0 = 19,0$.

$\sum_{\text{по}(-)} |\text{стр} 3 T_1| \rightarrow 3.2 T_3$ — просуммировать отрицательные числа строки 3 табл. 1 и результат записать в клетку 3.2 табл. 3.

Например, для рис. 1: $-6,0 - 4,0 = -10,0$.

$\sum_{\substack{\text{соотв} \\ (+) \text{стр} 3}} |\text{стр} 4 T_1| \rightarrow 4.1 T_3$ — просуммировать числа строки 4 табл. 1, находящиеся в одних графах с положительными числами строки 3, и результат записать в клетку 4.1 табл. 3. Другими словами, суммируемые в заданных пределах числа строки 4 соответствуют положительным числам строки 3.

Например, для рис. 1: $1,0 - 6,0 = -5,0$.

$\sum_{\substack{\text{соотв} \\ (-) \text{стр} 3}} |\text{стр} 4 T_1| \rightarrow 4.2 T_3$ — та же операция суммирования, но над числами строки 4, соответствующими отрицательным числам строки 3.

Например, для рис. 1: $5,0 + 20,0 = 25,0$.

Нахождение и запись максимальных (минимальных) чисел

$$\max \text{ из } |\text{стр } \alpha|_1^n T_1 \rightarrow \alpha.m_1 T_2$$

— найти максимальное число в строке α табл. 1 и записать его в клетку $\alpha.m_1$ табл. 2.

$$\min \text{ из } |\text{стр } \alpha|_1^n T_1 \rightarrow \alpha.m_2 T_2$$

— найти минимальное число из строки α табл. 1 и записать его в клетку $\alpha.m_2$ табл. 2.

Обозначение *absmax* (*absmin*) указывает на то, что максимальное (минимальное) число берется по абсолютному значению.

Например, для $|\text{стр } 4|_1^4 T_1$ из рис. 1 максимальное число равно 20,0; минимальное число равно $-6,0$, максимальное и минимальное по абсолютному значению числа соответственно равны 20,0 и 1,0.

Выборочное суммирование и определение максимальных (минимальных) величин могут быть использованы, в частности, для составления программы вычислений в так называемой таблице расчетных усилий.

Умножение (деление)

$$| \text{стр } \alpha T_A |_{n_1}^{+} \times | \text{стр } \beta T_B |_{n_2}^{+} \rightarrow | \text{стр } \gamma T_B |_{n_3}^{+}$$

— совместить строку α табл. A и строку β табл. B , начиная с клеток $\alpha.n_1 T_A$ и $\beta.n_2 T_B$, и попарные произведения рядом расположенных чисел записать в строку γ табл. B , начиная с клетки $\gamma.n_3 T_B$. Интервал умножения равен $\nu+1$ клеток.

Например, если $\alpha=1$, $\beta=3$, $\gamma=1$, $n_1=1$, $n_2=3$, $n_3=5$, $\nu=4$, то получим

$$| \text{стр } 1 T_A |_1^+ \times | \text{стр } 3 T_B |_3^+ \rightarrow | \text{стр } 1 T_B |_5^+$$

или в числах для рис. 3 (номера граф указаны над строками $1T_A$ и $1T_B$ и под строкой $3T_B$) получим $1,0 \times 2,0 = 2,0$; $2,0 \times 3,0 = 6,0$; $3,0 \times (-1,0) = -3,0$; $2,0 \times 2,0 = 4,0$; $4,0 \times 1,0 = 4,0$.

Чаще всего встречается, что пределы множимого, множителя и произведения совпадают, то есть $n_1=n_2=n_3$. В этом случае пределы ставятся только в левой части равенства. Например,

	1	2	3	4	5
Стр 1 T_A	1,0	2,0	3,0	2,0	4,0
Стр 3 T_B	2,0	3,0	-1,0	2,0	1,0
	3	4	5	6	7

	5	6	7	8	9
Стр 1 T_B	2,0	6,0	-3,0	4,0	4,0

Рис. 3

	1	2	3	4	5	
1	1,0	2,0	5,0	6,0	7,0	из табл. 1
2	3,0	4,0	-1,0	-2,0	2,0	из табл. 7
3	3,0	6,0	-5,0	-12,0	14,0	из табл. 8
7	8,0					из табл. 8

Рис. 4

если $\alpha=1$, $\beta=2$, $\gamma=3$, $A=1$, $B=7$, $B=8$, $n_1=n_2=n_3=1$ и $\nu=4$, то получим

$$| \text{стр } 1 T_1 |_1^+ \times | \text{стр } 2 T_2 |_1^+ \rightarrow | \text{стр } 3 T_3 |_1^+$$

или в числах для рис. 4: $1,0 \times 3,0 = 3,0$; $2,0 \times 4,0 = 8,0$; $5,0 \times (-1,0) = -5,0$; $6,0 \times (-2,0) = -12,0$; $7,0 \times 2,0 = 14,0$.

$$| \text{гра } \alpha T_A |_{m_1}^{+} \times | \text{гра } \beta T_B |_{m_2}^{+} \rightarrow | \text{гра } \gamma T_A |_{m_3}^{+}$$

— совместить графу α табл. A и графу β табл. B , начиная с клеток $m_1.\alpha T_A$ и $m_2.\beta T_B$, и попарные произведения рядом

расположенных чисел записать в графу γ табл. B , начиная с клетки $m_3 \cdot \gamma T_B$. Интервал умножения равен $\gamma + 1$ клеток.

Например, если $\alpha = 1, \beta = 3, \gamma = 1, A = 1, B = 2, B = 3, m_1 = m_2 = m_3 = 1$ и $\gamma = 5$, то получим

$$| \text{гр } 1 T_1 \times \text{гр } 3 T_2 |_1^5 \rightarrow \text{гр } 1 T_3,$$

или в числах для рис. 5 $1,0 \times 3,0 = 3,0; 2,0 \times 4,0 = 8,0; 5,0 \times (-1,0) = -5,0; 7,0 \times 2,0 = 14,0; 4,0 \times 3,0 = 12,0; -6,0 \times 5,0 = -30,0$.

$$| \text{стр } \alpha T_A |_{n_1}^{+\gamma} \times | \text{гр } \beta T_B |_{m_1}^{+\gamma} \rightarrow | \text{стр } \gamma T_B |_{n_1}^{+\gamma}$$

— умножить числа строки α табл. A , начиная с клетки αn_1 ,

	T_1	T_2		T_3		T_5
	1	3		1		3
1	1,0	3,0		3,0		2,0 (15,0)
2	2,0	4,0		8,0		
3	5,0	-1,0		-5,0		
4	7,0	2,0		14,0		
5	4,0	3,0		12,0		
6	-6,0	5,0		-30,0		
7	2,0	3,0				
8	—	4,0				

Рис. 5.

на числа графы β табл. B , начиная с клетки $m_1 \cdot \beta$, и результаты записать в строку γ табл. B , начиная с клетки $\gamma \cdot n_3$. Интервал умножения равен $\gamma + 1$ клеток (как в строке, так и графе).

Последняя программа расшифровывается следующим образом:

$$\alpha \cdot n_1 T_A \times m_1 \cdot \beta T_B \rightarrow \gamma \cdot n_3 T_B;$$

$$\alpha \cdot (n_1 + 1) T_A \times (m_1 + 1) \cdot \beta T_B \rightarrow \gamma \cdot (n_3 + 1) T_B;$$

$$\alpha \cdot (n_1 + 2) T_A \times (m_1 + 2) \cdot \beta T_B \rightarrow \gamma \cdot (n_3 + 2) T_B;$$

$$\dots \dots \dots$$

$$\alpha \cdot (n_1 + \gamma) T_A \times (m_1 + \gamma) \cdot \beta T_B \rightarrow \gamma \cdot (n_3 + \gamma) T_B.$$

или в числах для рис. 6: $1,0 \times 2,0 = 2,0$; $3,0 \times 4,0 = 12,0$; $-2,0 \times 3,0 = -6,0$; $\dots$ $4,0 \times (-3,0) = -12,0$.

Читатель без труда расшифрует также выражения:

$$|\text{стр } \alpha T_A|_{n_1}^{+v} \times |\text{стр } \beta T_B|_{n_2}^{+v} \rightarrow |\text{гр } \gamma T_B|_{m_3}^{+v};$$

$$|\text{гр } \alpha T_A|_{m_1}^{+v} \times |\text{гр } \beta T_B|_{m_2}^{+v} \rightarrow |\text{стр } \gamma T_B|_{n_3}^{+v};$$

$$|\text{гр } \alpha T_A|_{m_1}^{+v} \times |\text{стр } \beta T_B|_{n_2}^{+v} \rightarrow |\text{стр } \gamma T_B|_{n_3}^{+v} \text{ и т. д.}$$

T_A						T_B	
	n_1	n_1+1	n_1+2	...	n_1+v		β
α	1,0	3,0	-2,0		4,0	m_1	2,0
						m_1+1	4,0
						m_1+2	3,0
						..	...
						m_1+v	-3,0

T_B					
	n_3	n_3+1	n_3+2	...	n_3+v
γ	2,0	12,0	-6,0	...	-12,0

Рис. 6.

Операция умножения строки на графу (графы на строку) имеет то неудобство, что нельзя совмещать перемножаемые элементы, так как клетка таблицы обычно представляет собой вытянутый прямоугольник. Если клетки сделать квадратными, то совмещение возможно, однако приходится поворачивать одну из таблиц, что затрудняет работу и нередко приводит к ошибкам. Поэтому при составлении алгоритмов и программ для операторов клавишных автоматов следует по возможности избегать этой операции. Большей частью удастся, записывая попеременно результаты в одних таблицах в строки, в других — в графы, заменить операцию умножения строки на графу операцией умножения строк (или граф).

Если в левых частях приведенных выше программ вместо знака умножения ($\times$) стоит знак деления ($/$), то выполняется деление соответствующих чисел.

Операции умножения (деления) группы чисел на одно число записываются так:

$$|\text{стр } \alpha|_{n_1}^{+v} \times \beta \cdot n_2 \rightarrow |\text{стр } \gamma|_{n_3}^{+v};$$

$$|\text{стр } \alpha|_{n_1}^{+v} / n_2 \cdot \beta \rightarrow |\text{гр } \gamma|_{m_3}^{+v}.$$

Эти операции представляют собой частные случаи рассмотренных более сложных операций и дополнительных пояснений не требуют.

Суммирование попарных произведений (частных)

Если в левой части приведенной выше программы умножения стоит также знак суммирования Σ , а справа — обозначение клетки (то есть адрес одного числа), то это значит, что следует выполнить суммирование попарных произведений (частных) *. Ниже приведен ряд таких программ.

$$\Sigma (|\text{стр } \alpha T_A|_{n_1}^{+\nu} \times |\text{стр } \beta T_B|_{n_2}^{+\nu}) \rightarrow \gamma \cdot n_3 T_V.$$

Например, если $A=1$, $B=7$, $V=8$, $n_1=n_2=n_3=1$, $\alpha=1$, $\beta=2$, $\gamma=7$, $\nu=4$, то

$$\Sigma |\text{стр } 1 T_1 \times \text{стр } 2 T_7|_1^5 \rightarrow 7 \cdot 1 T_8,$$

или в числах для рис. 4: $1,0 \times 3,0 + 2,0 \times 4,0 + 5,0 \times (-1,0) + 6,0 \times (-2,0) + 7,0 \times 2,0 = 8,0$.

$$\Sigma (|\text{гр } \alpha T_A|_{m_1}^{+\nu} \times |\text{гр } \beta T_B|_{m_2}^{+\nu}) \rightarrow m_3 \cdot \gamma T_V.$$

Например, если $A=1$, $B=2$, $V=5$, $m_1=m_2=m_3=1$, $\alpha=1$, $\beta=3$, $\gamma=3$, $\nu=5$, то

$$\Sigma |\text{гр } 1 T_1 \times \text{гр } 3 T_2|_1^6 \rightarrow 1 \cdot 3 T_5,$$

или в числах для рис. 5: $1,0 \times 3,0 + 2,0 \times 4,0 + 5,0 \times (-1,0) + 7,0 \times 2,0 + 4,0 \times 3,0 - 6,0 \times 5,0 = 2,0$.

Аналогично реализуется программа суммирования произведений элементов строки и графы (графы и строки).

$$\Sigma (|\text{стр } \alpha T_A|_{n_1}^{+\nu} \times |\text{гр } \beta T_B|_{m_1}^{+\nu}) \rightarrow m_3 \cdot \gamma T_V;$$

$$\Sigma (|\text{гр } \alpha T_A|_{m_1}^{+\nu} \times |\text{стр } \beta T_B|_{n_1}^{+\nu}) \rightarrow \gamma \cdot m_3 T_V.$$

Суммирование попарных произведений по диагонали

В ряде случаев (в частности при численном интегрировании) целесообразно пользоваться особой операцией суммирования попарных произведений по диагоналям.

$$\Sigma |\text{гр } 1 T_1 \times \text{гр } 3 T_2|_{\text{диаг}}^6 \rightarrow 1 \cdot 3 T_5$$

* Накопление сумм произведений (частных) выполняется на клавишных машинах автоматически.

— совместить графу 1 табл. 1 с графой 3 табл. 2 и сумму парных произведений чисел по диагоналям (крест—накрест между жирными линиями) записать в клетку 1.3 табл. 5. Пределы суммирования от первой до шестой строки, например, для рис. 5 $(1,0 \times 4,0 + 3,0 \times 2,0) + (5,0 \times 2,0 - 1,0 \times 7,0) + (4,0 \times 5,0 - 3,0 \times 6,0) = 15,0$. Результат записать в клетке 1.3 табл. 5 (в скобках).

Некоторые из рассмотренных выше программ описывают матричные операции линейной алгебры (операции над векторами), но благодаря введению пределов этим операциям сообщается полная определенность. Действительно, сокращенные обозначения, применяемые в линейной алгебре, не могут быть непосредственно использованы в инженерных программах, так как в них отсутствует информация о том, какой вектор изображается данным символом (вектор-столбец или вектор-строка), к какой таблице он относится и каковы его размеры. Достаточная информация содержится лишь в полном изображении векторов, а такие изображения ввиду их громоздкости неудобно использовать в программах.

Симметрия и взаимность

Клетки, расположенные симметрично относительно главной диагонали таблицы (см. рис. 2), обозначаются одинаковыми цифрами и отличаются только их порядком расположения. Например, симметрично клетке 2.5T₂ располагается клетка 5.2T₂. Цифры 2 и 5 меняются местами.

Симметричные клетки таблицы условимся называть взаимными в том случае, если в них записано одно и то же число.

Операция засылки во взаимные клетки:

2.5T₂ → 5.2T₂ — взять число из клетки 2.5 табл. 2 и записать его в клетку 5.2 той же таблицы.

Для рис. 2 это число равно 1,0.

Условимся также называть взаимными одноименные графу и строку, если они состоят из взаимных клеток.

Операция засылки:

стр 2T₂ → гр 2T₂ — взять числа из строки 2 табл. 2 и записать их в симметричные клетки графы 2 табл. 2. Например, для рис. 2: 2.1 → 1.2; 2.3 → 3.2, 2.4 → 4.2 и т. д.

Запись простейших программ в заглавной строке таблицы

Для несложных расчетов, выполняемых в одной таблице, удобно использовать обычную запись последовательности действий, указанную на рис. 7.

Обозначение (1) × (2) в графе 5 указывает, что числа графы 1 умножаются на соответствующие числа графы 2 и результаты записываются в графу 5. Обозначение (4) : (6) в графе 7 указывает, что числа графы 4 делятся на соответствующие чи-

сла графы 6 и результаты записываются в графу 7. Обозначение (5) + (6) + (7) в графе 8 указывает, что числа граф 5, 6 и 7 суммируются и результаты записываются в графу 8.

		(1) × (2)		(3) × (4)		(4) · (6)		(5) + (6) + (7)	
	1	2	3	4	5	6	7	8	
1									
2									
3									

Рис. 7.

Указанные обозначения хотя и просты, но расплывчаты. С их помощью нельзя указать точный адрес числа или в заданных пределах группы чисел в разных таблицах, а без этого трудно составить программы для сложных расчетов.

Сочетания элементарных программ

Более сложные программы образуются сочетанием рассмотренных элементарных выражений.

В виде примера рассмотрим операцию численного интегрирования эпюр усилий. Для двух прямолинейных эпюр моментов (рис. 8) формула численного интегрирования для определения перемещения δ_{ik} имеет следующий вид:

$$\delta_{ik} = \int \frac{M_i M_k dS}{EI} = \frac{L}{6EI} [2(a_1 b_1 + a_2 b_2) + (a_1 b_2 + a_2 b_1)].$$

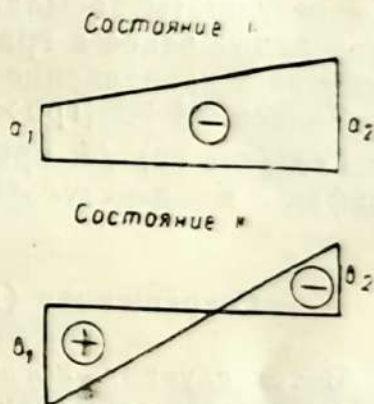


Рис. 8.

В правой части формулы первый член представляет собой удвоенную сумму попарных произведений ординат эпюры моментов, второй — сумму произведений тех же ординат по диагоналям. В общем случае эпюра моментов состоит из ряда прямолинейных участков (криволинейная эпюра аппроксимируется ломаной линией).

Если поместить ординаты a (состояние i) в графу a табл. 1 и ординаты $\frac{bL}{6EI}$ (состояние k) в графу b табл. 2, то получим

следующую программу для определения перемещений:

$$2\Sigma | \text{гр } i T_1 \times \text{гр } k T_2 | + \Sigma | \text{гр } i T_1 \times \text{гр } k T_2 |_{\text{линт}} \rightarrow i. k T_A.$$

Левая часть программы образована сочетанием двух элементарных программ; в правой части указан адрес вычисленного коэффициента системы уравнений табл. А.

Запись эскизных программ для электронных вычислительных машин

Запись эскизных программ производится так же, как и рассмотренных выше инженерных программ для клавишных автоматов, но учитывается, что оперативная память электронных вычислительных машин ограничена и поэтому следует по возможности использовать одни и те же ячейки памяти для записи исходных, промежуточных и окончательных чисел.

Моделью памяти в эскизных программах служит таблица (матрица) и ее элементы — строки, графы, клетки. Следовательно, экономное использование памяти выражается в записи чисел в минимальном количестве элементов таблицы; при этом имитируется характерный для ЭВМ процесс вытеснения одних чисел (ставших ненужными) другими в одних и тех же ячейках, в данном случае клетках таблицы.

Например, запись $\text{гр } c \leftarrow a \times \text{гр } b \rightarrow \text{гр } c$ обозначает, что из чисел графы c вычитаются умноженные на постоянную a числа графы b и результаты засылаются в ту же графу c . При этом числа, записанные ранее в графе c , автоматически стираются. Стрелка указывает направление движения программы слева направо.

Запись $i.k + \Sigma | \text{гра } a \times \text{гр } b |_1^n \rightarrow i.k$ обозначает, что к числу в ячейке (клетке) $i.k$ прибавляется сумма произведений чисел в графах a и b и результат засылается в ту же ячейку $i.k$.

Сокращенная (координатная) запись программ

В тех случаях, когда последовательные выражения в программе имеют одну и ту же структуру и отличаются лишь адресами, можно применить сокращенную координатную запись программы, указывая закономерность изменения входящих в программу переменных величин (адресов). Например,

$$\Sigma | \text{гр } i T_A \times \text{гр } k T_B |_1^m \rightarrow i. k T_B \rightarrow k. i T_B$$

для $i=1$ до n ,
 $k=i$ до S .

Проверка: $\Sigma | \text{стр } i \text{ } T_B |_1^n \rightarrow i. ST_B$. Программа выполняется для всех указанных значений индексов i и k , причем каждому значению предыдущего индекса i соответствует полный цикл изменения последующего индекса k .

Развернутая запись той же программы имеет вид:

$$\begin{array}{ll}
 \Sigma | \text{гр } 1 \text{ } T_A \times \text{гр } 1 \text{ } T_B |_1^m \rightarrow 1.1 \text{ } T_B & k=1 \\
 \Sigma | \text{гр } 1 \text{ } T_A \times \text{гр } 2 \text{ } T_B |_1^m \rightarrow 1.2 \text{ } T_B \rightarrow 2.1 \text{ } T_B & k=2 \\
 \dots & \dots \\
 \Sigma | \text{гр } 1 \text{ } T_A \times \text{гр } n \text{ } T_B |_1^m \rightarrow 1.n \text{ } T_B \rightarrow n.1 \text{ } T_B & k=n \\
 \Sigma | \text{гр } 1 \text{ } T_A \times \text{гр } ST_B |_1^m \rightarrow 1. ST_B & k=S
 \end{array} \left. \vphantom{\begin{array}{l} k=1 \\ k=2 \\ \dots \\ k=n \\ k=S \end{array}} \right\} i=1$$

Проверка: $\Sigma | \text{стр } 1 \text{ } T_B |_1^n \rightarrow 1. ST_B$

$$\begin{array}{ll}
 \Sigma | \text{гр } 2 \text{ } T_A \times \text{гр } 2 \text{ } T_B |_1^m \rightarrow 2.2 \text{ } T_B & k=2 \\
 \Sigma | \text{гр } 2 \text{ } T_A \times \text{гр } 3 \text{ } T_B |_1^m \rightarrow 2.3 \text{ } T_B \rightarrow 3.2 \text{ } T_B & k=3 \\
 \dots & \dots \\
 \Sigma | \text{гр } 2 \text{ } T_A \times \text{гр } n \text{ } T_B |_1^m \rightarrow 2.n \text{ } T_B \rightarrow n.2 \text{ } T_B & k=n \\
 \Sigma | \text{гр } 2 \text{ } T_A \times \text{гр } ST_B |_1^m \rightarrow 1. ST_B & k=S
 \end{array} \left. \vphantom{\begin{array}{l} k=2 \\ k=3 \\ \dots \\ k=n \\ k=S \end{array}} \right\} i=2$$

Проверка: $\Sigma | \text{стр } 2 \text{ } T_B |_1^n \rightarrow 2. ST_B$

и т. д. до n

$$\begin{array}{ll}
 \Sigma | \text{гр } n \text{ } T_A \times \text{гр } n \text{ } T_B |_1^m \rightarrow n.n \text{ } T_B & k=n \\
 \Sigma | \text{гр } n \text{ } T_A \times \text{гр } ST_B |_1^m \rightarrow n. ST_B & k=S
 \end{array} \left. \vphantom{\begin{array}{l} k=n \\ k=S \end{array}} \right\} i=n$$

Проверка: $\Sigma | \text{стр } n \text{ } T_B |_1^n \rightarrow n. ST_B$

Блочная запись программ

Типовые программы даже различных классов сооружений имеют между собой ряд общих элементов, которые можно выделить в стандартные блоки, более или менее крупные. Стандартными блоками-программами описываются, в частности, известные матричные операции [67].

Для обозначения блоков применим наглядные символы, которые, как показала практика, легко усваиваются операторами.

Ниже помещен ряд стандартных блоков-программ в блочной и сокращенной координатной записи. Пределы не указаны, так

как имеются в виду операции над всеми элементами таблиц. Под обозначением $\text{гр} T_A$ ($\text{стр} T_A$) — понимается совокупность всех граф (строк) таблицы A . Об использовании пределов для описания операций над частями таблиц будет сказано дальше.

Блок 1.

$$\begin{array}{ccc} \text{гр} T_A & \times & \text{гр} \beta T_B \rightarrow \text{гр} T_B. \\ (\text{стр}) & (\text{стр}) & (\text{стр}) \end{array}$$

В скобках указаны возможные варианты блока; замена обозначений может быть произведена в любых комбинациях в одном или нескольких членах. При этом размеры перемножаемых элементов (строк, граф) должны быть одинаковы.

Сокращенная координатная запись блока:

$$\begin{array}{c} \text{гр} iT_A \times \text{гр} \beta T_B \rightarrow \text{гр} iT_B, \\ i=1 \text{ до } S. \end{array}$$

Проверка: $\Sigma | \text{стр} iT_B |_1^n \rightarrow i. ST_B$.

Если справа стоит обозначение $\text{стр} iT_B$, то проверки имеют вид: $\Sigma | \text{гр} iT_B |_1^n \rightarrow S. iT_B$.

Блок 2.

$$\begin{array}{ccc} \Sigma | \text{гр} T_A \times \text{гр} \beta T_B | & \rightarrow & \text{гр} \gamma T_B. \\ (\text{стр}) & (\text{стр}) & (\text{стр}) \end{array}$$

Сокращенная координатная запись блока:

$$\begin{array}{c} \Sigma | \text{гр} iT_A \times \text{гр} \beta T_B | \rightarrow i. \gamma T_B, \\ i=1 \text{ до } S^*. \end{array}$$

Проверка: $\Sigma | \text{гр} \gamma T_B |_1^n \rightarrow S. \gamma T_B$.

Если справа стоит обозначение γiT_B , то будем иметь проверку $\Sigma | \text{стр} \gamma T_B |_1^n \rightarrow \gamma. ST_B$.

Блок 3.

$$\begin{array}{ccc} \Sigma | \text{гр} T_A \times \text{гр} T_B | & \rightarrow & \text{гр} T_B. \\ (\text{стр}) & (\text{стр}) & (\text{стр}) \end{array}$$

Сокращенная координатная запись блока:

$$\begin{array}{c} \Sigma | \text{гр} iT_A \times \text{гр} kT_B | \rightarrow k. iT_B \\ \text{для } i=1 \text{ до } n, k=1 \text{ до } S. \end{array}$$

* Графа (строка) S табл. A является суммарной графой (строкой), вычисляемой для последующего контроля.

$$\Sigma | \text{стр} T_A |_1^n \rightarrow i. ST_A; \quad \Sigma | \text{гр} iT_A |_1^m \rightarrow S. iT_A.$$

Проверка: $\Sigma | \text{гр } i |_1^n T_B \rightarrow S. i T_B.$

Если справа стоит обозначение $i.k$, то проверки примут вид:

$$\Sigma | \text{стр } i |_1^n T_B \rightarrow i. S T_B.$$

Блок 4.

$$\Sigma | \text{гр } T_A \times \text{гр } T_B | \rightarrow T_{\text{всн}}.$$

(стр) (стр)

Значок «сим» указывает, что табл. B является симметричной, и это учитывается в программе.

Сокращенная координатная запись блока 4 приведена на стр. 26.

Отличие рассмотренных стандартных блоков от соответствующих матричных формул состоит в указании, с какими элементами (строками или графами) оперирует данный блок. Благодаря этому отпадает необходимость в некоторых формальных действиях, к которым приходится прибегать в матричном исчислении.

За основу в матричном анализе принята операция умножения матриц $A \times B = B$, которая состоит в суммировании произведений строк матрицы A на графы матрицы B с записью результатов в строки матрицы B . Далее вводится понятие транспонированной матрицы A' , которая получается из данной матрицы A путем замены строк графами.

Не говоря о том, что операция умножения строк на графы неудобна для оператора клавишной машины, указанная символика приводит к тому, что при описании других операций приходится выполнять лишние действия. Например, запись программы блока 3

$$\Sigma | \text{гр } T_A \times \text{гр } T_B | \rightarrow \text{гр } T_B,$$

весьма удобного для вычислений и часто применяемого, в матричной форме имеет вид:

$$A' \times B = B_1;$$

$$B'_1 = B.$$

Операции транспонирования являются излишними.

Как видим, аппарат матричного исчисления, весьма эффективный при математических исследованиях, не всегда целесообразно применять в неизменном виде для выполнения расчетов на клавишных машинах.

Требуется вносить коррективы, учитывая особенности и возможности вычислительных машин.

Блочная запись программ (с указанием пределов)

Недостаток рассмотренной выше блочной записи состоит в ее неопределенности, так как в обозначениях таблиц не указаны их размеры.

Пользоваться такой записью можно только в тех случаях, когда размеры таблиц приведены в соответствие так, чтобы строки и графы, участвующие в операциях, имели одинаковое количество клеток. В матричном исчислении это приводит к тому, что одну из матриц дополняют нулевыми элементами. Если же требуется выполнять операции над частями таблиц, то последние нужно переписывать в другие бланки и сообщать им номера как новым таблицам. Такое переписывание не только занимает время, но и нередко приводит к ошибкам из-за описок. Эти недостатки можно устранить, если пользоваться координатной записью блока с указаниями пределов в обозначениях элементов и закономерностей изменения адресов внутри блока. Этим неопределенность полностью устраняется.

В виде примера рассмотрим блок 3 для частей таблиц A и B , обведенных жирными линиями на рис. 9. В графе S_1 даны суммы чисел рассматриваемой части табл. A .

$$\Sigma(|\text{гр } i T_B|_3^6 \times |\text{гр } k T_A|_2^5) \rightarrow (k-1) \cdot (i-5) T_B$$

для $i=6$ до 9, $k=2$ до S_1 .

Принимая $i=6$, $k=2$, получаем

$$\Sigma(|\text{гр } 6 T_B|_3^6 \times |\text{гр } 2 T_A|_2^5) \rightarrow 1.1 T_B.$$

В числах: $2,0 \times 1,0 + 1,0 \times 2,0 + 3,0 \times 2,0 + 2,0 \times 3,0 = 16,0$.

Принимая $i=6$, $k=3$,

$$\Sigma(|\text{гр } 6 T_B|_3^6 \times |\text{гр } 3 T_A|_2^5) \rightarrow 2.1 T_B.$$

В числах: $2,0 \times 2,0 + 1,0 \times 3,0 + 3,0 \times 2,0 + 2,0 \times 4,0 = 21,0$.

Согласно программе заполнение табл. B производится по графно.

Результаты в числовом виде приведены в табл. B .

Проверка: в строке S_1 табл. B содержатся суммы чисел в графах.

Рассмотрим еще видоизменение блока 3, когда строки табл. B умножаются на графы табл. A и заполнение табл. G производится построчно.

Этот вариант для выделенных на рис. 9 частей таблиц имеет следующий вид:

$$\Sigma(|\text{стр } i T_B|_6^9 \times |\text{гр } k T_A|_2^5) \rightarrow (i-2) \cdot (k-1) T_G$$

для $i=3$ до 6, $k=2$ до 6.

Принимая $i=3$, $k=2$, получим

$$\Sigma(|\text{стр } 3 \text{ } T_B|_6^9 \times |\text{гр } 2 \text{ } T_A|_2^5) \rightarrow 1.1 \text{ } T_G.$$

В числах: $2,0 \times 1,0 + 3,0 \times 2,0 + 2,0 \times 2,0 + 1,0 \times 3,0 = 15,0$.

Принимая $i=3$, $k=3$, получаем

$$\Sigma(|\text{стр } 3 \text{ } T_B|_6^9 \times |\text{гр } 3 \text{ } T_A|_2^5) \rightarrow 1.2 \text{ } T_G.$$

В числах: $2,0 \times 2,0 + 3,0 \times 3,0 + 2,0 \times 2,0 + 1,0 \times 4,0 = 21,0$ и т. д.

Результаты в числовом виде приведены в табл. Г. Проверка: в графе S_1 табл. Г содержатся суммы чисел в строках.

T_A										
1	2	3	4	5	6	7	...	n	s	s_1
1										
2	1,0	2,0	3,0	1,0	4,0					11,0
3	2,0	3,0	1,0	2,0	4,0					12,0
4	2,0	2,0	3,0	1,0	2,0					10,0
5	3,0	4,0	2,0	1,0	3,0					13,0
...										
m										

T_B										
...	6	7	8	9	10	...	n			
1										
2										
3	2,0	3,0	2,0	1,0						
4	1,0	4,0	1,0	2,0						
5	3,0	1,0	3,0	2,0						
6	2,0	3,0	4,0	2,0						
...										

T_E				
1	2	3	4	
1	16,0	22,0	22,0	15,0
2	21,0	32,0	29,0	20,0
3	20,0	22,0	24,0	15,0
4	9,0	15,0	11,0	9,0
5	24,0	39,0	30,0	22,0
s_1	90,0	130,0	116,0	81,0

T_G					
1	2	3	4	5	s_1
1	15,0	21,0	17,0	11,0	27,0
2	17,0	24,0	14,0	12,0	28,0
3	17,0	23,0	23,0	10,0	28,0
4	22,0	29,0	25,0	14,0	34,0

Рис. 9.

Выше были рассмотрены наиболее сложные случаи произвольного расположения умножаемых числовых массивов в полях таблиц.

Обычно на практике приходится иметь дело с более упорядоченным расположением массива, когда он занимает, например, все поле таблицы по вертикали и часть поля по горизонтали (пунктирные линии на рис. 9).

Основное внимание в данном разделе уделено правилам использования адресного языка Института кибернетики АН УССР для записи алгоритмов, характерных для инженерных расчетов.

Адресные программы, как и рассмотренные выше инженерные программы, построены на адресном принципе, но отличаются тем, что язык их почти полностью приближен к машинному.

Если закономерность изменения адресов в инженерных программах воспринимается человеком-вычислителем как нечто очевидное, то в адресных программах адреса изменяются с помощью счета, а управление переходом от одних вычислительных этапов к другим осуществляется, как это делается в ЭВМ, программно в зависимости от выполнения тех или иных логических условий.

Логических действий в адресных программах обычно значительно больше, чем чисто арифметических. В разработке логической схемы счета и состоит своеобразие адресных программ. Ниже дана краткая сводка основных обозначений, применяемых в адресных программах.

$'a=b$ —содержимое ячейки a (штрих слева означает «содержимое») равно b . То же понятие можно перефразировать так—число b содержится в ячейке a ;

a —называется адресом первого ранга числа b .

Если, однако, b представляет собой не число, а адрес некоторого числа c , то есть $'b=c$, то подставив вместо b равнозначную ей величину $'a$, получаем $'('a)=c$ или $''a=c$, или $^2a=c$.

a —называется адресом второго ранга числа c . Это понятие выражает ту мысль, что в a содержится адрес, в котором уже содержится число c . Если c будет адресом некоторого числа d , $'c=d$, то получим

$$^3a = ''('a) = ^2b = 'c = d.$$

c —адрес первого ранга числа d ;

b —адрес второго ранга числа d ;

a —адрес третьего ранга числа d .

В общем случае адрес n -го ранга числа d выражается через адрес $n-1$ -го ранга так:

$$^na = '^{'^{n-1}a}.$$

Операция засылки числа в некоторую ячейку обозначается стрелкой

$$\boxed{'b \rightarrow c}$$

Содержимое ячейки b заслать в ячейку c с вытеснением ее предыдущего содержимого.

После выполнения этой операции соблюдается равенство

$$'b = 'c,$$

то есть содержимые ячеек b и c равны между собой.

Операции сложения, вычитания, умножения, деления изображаются общепринятыми символами $+$, $-$, $\times$, $:$:

$$'d_0 + 'm_1 + 'm = b,$$

d_0 , m_1 и m представляют собой адреса первого ранга, сумма содержимых которых равна b .

$$\boxed{'d_0 + 'm_1 + 'm \rightarrow a}$$

Это выражение имеет иной смысл, чем предыдущее — сумма содержимых ячеек d_0 , m_1 и m засылается в ячейку a . После этого будет соблюдаться равенство

$$'d_0 + 'm_1 + 'm = 'a.$$

П р а в и л о: для того, чтобы стрелку заменить знаком равенства, необходимо в правой части прибавить штрих.

Адресные формулы, заменяющие команды в программах для ЭВМ, состоят из двух частей: в левой части указаны адреса чисел, участвующих в операции, в правой — адреса результатов.

Выражения, обведенные рамками, представляют собой простейшие адресные формулы.

Рассмотрим теперь более сложные выражения, с которыми придется встречаться в дальнейшем.

$$'('d_0 + 'm_1 + 'm) = "a$$

— слева записан адрес второго ранга $"a$ при условии, что

$$'d_0 + 'm_1 + 'm = 'a.$$

В одном выражении могут присутствовать адреса разных рангов. Например, если адрес m_1 содержится в ячейке m_2 ($m_1 = 'm_2$), то получим

$$'('d_0 + 'm_1 + 'm) = '('d_0 + "m_2 + 'm).$$

Если к тому же адрес m равен сумме содержимых ячеек m_3 и m_4 ($m = 'm_3 + 'm_4$), то $'('d_0 + 'm_1 + 'm) = '['d_0 + "m_2 + '('m_3 + 'm_4)]$.

Расшифровка сложных адресных формул производится путем последовательного снижения рангов и раскрытия скобок.

П р а в и л о: раскрывать скобки можно лишь после того, как будет вычислено стоящее в них выражение. Скобки раскрываются в последовательности: сначала круглые, затем квадратные, затем фигурные и т. д.

Например выражение

$$' \{ ['d_0 + "m_2 + ' ('m_3 + 'm_4)] + {}^3m_5 \} = a_2$$

раскрывается следующим образом:

$$'m_3 + 'm_4 = m;$$

$$'d_0 + "m_2 + 'm = a;$$

$$'a + {}^3m_5 = a_1;$$

$$'a_1 = a_2 *.$$

Представление формул в адресах высоких рангов значительно сокращает запись и облегчает обозрение формул.

При переводе на язык ЭВМ адресные формулы расшифровываются, причем промежуточные результаты засылаются в рабочие ячейки (по возможности в одни и те же).

Адреса высоких рангов должны быть последовательно снижены до первого ранга, для чего применяются операции сдвига их содержимых из одного адреса в другой и сложения команд.

В некоторых машинах имеется специальное устройство, так называемый регистр адреса, позволяющее значительно упростить операцию снижения ранга адреса, избежать сдвигов и сложений команд.

Приведем характерную адресную формулу, взятую из реальной программы в сокращенной и развернутой записи.

Сокращенная запись

$$"r - ' ('d_0 + 'm_1) \times ' ('d_0 + 'm_1 + 'm) : "d_0 \rightarrow 'r,$$

в развернутом виде (после раскрытия скобок) будет

$$'d_0 + 'm_1 \rightarrow r_1;$$

$$'r_1 + 'm \rightarrow r_2^{**};$$

$$"r - "r_1 \times "r_2 : "d_0 \rightarrow 'r.$$

Последняя формула выражает следующее: произвести действия умножения, деления, вычитания над содержимыми адресов второго ранга, указанными в левой части формулы, и результат направить в 'r — ячейку, адрес которой содержится в r.

* Выражение в фигурных скобках нельзя раскрывать путем простого сложения рангов, не принимая во внимание, где поставлены скобки. Сложение рангов возможно лишь в том случае, когда в скобках стоит только один символ:

$$' \{ ['^2(a)] \} = {}^4a.$$

** При этом содержимые ячеек r_1 и r_2 станут равны:

$$'r_1 = 'd_0 + 'm_1;$$

$$'r_2 = 'd_0 + 'm_1 + 'm.$$

Сокращенная запись более удобна, так как позволяет полностью обозреть формулу счета и сопоставить ее с алгебраическим выражением, которому она соответствует, что облегчает чтение и проверку программы.

Входящие в формулу счета адреса обычно являются переменными величинами.

Для исчерпывающего описания переменной величины необходимо знать ее начальное значение, закон изменения и область (или интервал) существования. Адрес ячейки, в которой хранится начальное значение переменной величины, будем называть ее фиксатором. Если переменной величиной является адрес, то, очевидно, фиксатором будет адрес второго ранга.

Область существования переменной величины (в частности, адреса) характеризуется некоторыми параметрами. Например, для матриц параметрами, определяющими область (интервал) изменения адресов, могут служить количество ячеек в строке (графе), порядок n квадратной матрицы, размеры m, n прямоугольной матрицы и т. п.

Адреса ячеек, в которых хранятся параметры, будем называть фиксаторами параметров.

Условимся обозначать фиксатор некоторой величины N этим же символом N .

Например, фиксатор матрицы A также обозначим символом A . Это значит, что в ячейке A хранится адрес первого (или предшествующего первому) элемента матрицы.

Фиксатор порядка матрицы n также обозначается n , то есть в ячейке n хранится порядок матрицы (n равно порядку матрицы) и т. д.

Закон изменения переменной величины в машине задается рядом дискретных значений. Если переменной величиной является адрес, то указывают разность между последующим и предыдущим его значениями, так называемый шаг переадресации.

Управление процессом изменения переменных величин (в данном случае адресов) осуществляется с помощью формул управления (или иначе предикатных формул, или формул условного перехода), которые передают управление в зависимости от выполнения определенных условий той или иной адресной формуле.

Формула управления (соответствующая команде управления для ЭВМ) в адресной программе записывается следующим образом:

$$p \{ \dots \} u, v$$

— проверить (p) условие, указанное в фигурных скобках; если это условие выполняется, управление передать формуле u , если нет — формуле v .

Условия предельного перехода, заключенные в фигурные скобки, могут быть $a=b$, $a \neq b$; $a \leq b$, $a > b$ и т. д.

Безусловная передача управления на формулу u записывается так:

адресная формула } u .

После выполнения очередной адресной формулы управление передается формуле u (безусловно, независимо от выполнения каких-либо условий).

С помощью формул управления даются указания об окончании данного этапа вычислений и переходе к следующему этапу, либо о прекращении процесса вычислений вообще (останове машины).

$p\{ \dots \} u$, ост

— проверить условие, указанное в фигурных скобках; если это условие выполняется, управление передать формуле u , если нет — прекратить вычисления (ост.).

Мы перечислили некоторые логические действия, которые в машинах реализуются по правилам математической логики*.

Укажем еще на одну логическую операцию, которая применяется не только при рабочем, но и адресном программировании, а именно, на операцию логического умножения, обозначаемую символом $\wedge$.

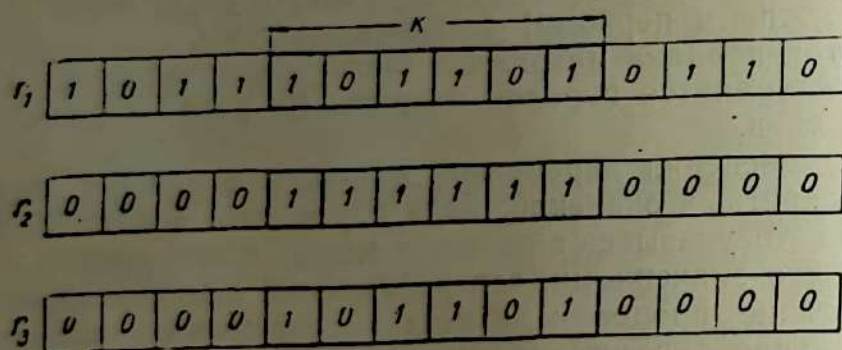


Рис. 10.

Логическое или поразрядное умножение, выполняемое по правилу $0 \wedge 0 = 0$; $0 \wedge 1 = 0$; $1 \wedge 0 = 0$; $1 \wedge 1 = 1$, применяется для выделения некоторого количества разрядов из ячейки памяти. В самом деле, для выделения из r_1 -ой ячейки k разрядов (рис. 10) достаточно логически умножить ее содержимое на со-

* Основные сведения по алгебре логики, необходимые при программировании, изложены в гл. II работы [26] или гл. I работы [8].

Увлекательное описание алгебры логики (или так называемой булевой алгебры — по имени английского математика Дж. Буля) и ее применения для построения математических машин можно найти в книге Э. Беркли «Символическая логика и разумные машины». Издательство иностранной литературы, М., 1961.

держимое ячейки r_2 , в которой в k разрядах помещены единицы, а в остальных — нули, и результат заслать в ячейку r_3 .

В ячейке r_3 получаем выделенную часть ячейки r_1 .

Указанная операция записывается на адресном языке следующим образом:

$$r_1 \wedge r_2 \rightarrow r_3$$

— логически умножить содержимое ячейки r_1 на содержимое ячейки r_2 и результат заслать в r_3 .

Адресная программа представляет собой набор следующих одна за другой формул, включающих операции формирования адресов, арифметического счета и управления (логических проверок).

Рассмотрим перевод некоторых простейших инженерных программ на язык адресного программирования.

Эскизная запись инженерной программы

$$\Sigma (|\text{стр } \alpha|_{n_1}^{+v} \times |\text{стр } \beta|_{n_2}^{+v}) \rightarrow r$$

выражает следующее: вычислить сумму произведений чисел в строках α и β в заданных пределах и результат суммирования заслать в ячейку r .

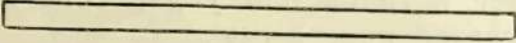
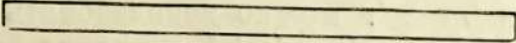
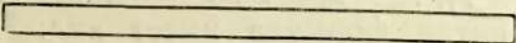
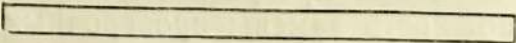
$n_1 + m$	Ячейки строк α (β)	$(n_2 + m)$
$n_1 + 0$		$(n_2 + 0)$
$n_1 + 1$		$(n_2 + 1)$
$n_1 + 2$		$(n_2 + 2)$
...	...	...
$n_1 + v$		$(n_2 + v)$

Рис. 11.

Из эскизной программы следует, что начальными адресами чисел строк α и β являются величины n_1 и n_2 , интервал изменения адресов строки равен v (количество ячеек строки минус единица). Согласно принятому правилу обозначим соответствующие фиксаторы этими же символами —

n_1 , n_2 и v .

Если шаг переадресации равен единице (то есть номер каждого последующего адреса отличается от предыдущего на единицу, как это показано на рис. 11), то адреса текущих ячеек строк α и β будут соответственно $n_1 + m$ и $n_2 + m$, где m — ячейка, на-

зывается счетчиком, содержимое которой последовательно изменяется на единицу и принимает значения $0, 1, 2, \dots, v$.

Теперь можно записать программу вычислений в адресном виде.

- 1) $0 \rightarrow r$;
- 2) $0 \rightarrow m$;
- 3) $'r + ('n_1 + 'm) \times ('n_2 + 'm) \rightarrow r$;
- 4) $'m + 1 \rightarrow m$;
- 5) $p \{ 'm \leq 'v \} 3, v$.

В первых двух формулах нули засылаются в ячейки r и m , так как начальное содержимое этих ячеек должно быть равно нулю.

В третьей формуле выполняется счет — накопление сумм произведений: к предыдущему содержимому ячейки r прибавляется произведение содержимых ячеек $('n_1 + 'm)$ и $('n_2 + 'm)$ и результат засылается в ячейку r (с вытеснением).

В четвертой формуле содержимое счетчика m изменяется на единицу.

В пятой формуле проверяется, не исчерпаны ли все числа строк. Если условие $'m \leq 'v$ выполняется (числа не исчерпаны), управление передается формуле 3 и счет продолжается, если нет (все числа строк исчерпаны), программа вычислений закончена и выполняется останов или управление передается формуле v другого отрезка программы.

Вместо условия $'m \leq 'v$ можно использовать условие совпадения величин $'m = 'v_1$, где $'v_1 = 'v + 1$ — полное количество ячеек в строке. Тогда пятая формула имеет вид:

- 5) $p \{ 'm = 'v_1 \} v, 3$.

При переводе адресной программы на язык ЭВМ следует иметь в виду возможность замены проверочных формул с тем, чтобы они соответствовали системе команд данной ЭВМ (например, в системе команд «Киева» имеется условие $'m \leq 'v$, а в системе команд БЭСМ 2М этого условия нет и его следует заменить условием $'m = 'v_1$).

Иногда фиксируют в ячейках n_1 и n_2 адреса ячеек, предшествующих первым, а под $'v$ понимают полное количество ячеек в строке (а не уменьшенное на единицу). Тогда изменится формула 2: $1 \rightarrow m$. Остальные формулы останутся без изменения.

Рассмотренная небольшая программа описывает элементарный цикл вычислений; количество циклов регламентируется формулой 5.

При переводе адресных программ на язык ЭВМ буквенные фиксаторы заменяются действительными адресами. Содержимые фиксаторов являются переменными и соответствуют характеристикам решаемой задачи.

Перейдем теперь к переводу более сложной эскизной программы, а именно — программы преобразования матриц к симметричному виду.

$$\Sigma | \text{gr } i \times \text{gr } k |_1^m A \rightarrow n_2, n_3 B;$$

$$i = n_1 + n_2, \quad k = n_1 + n_3.$$

n_1 — номер графы, предшествующей начальной графе табл. А (рис. 12)*.

n_2 и n_3 принимают значения

для $n_2 = 1$ до n ,

$n_3 = 1$ до q .

Значения $n_3 > n$ относятся к свободным членам.

Так как матрица B получается симметричной, то программой предусмотрено заполнение только половины ее (выше главной диагонали), далее идут свободные члены.

В матричных обозначениях указанную программу можно записать так: $A' \times A \rightarrow B$

— умножить матрицу A слева на транспонированную матрицу A' и результаты записать в матрицу B .

		T_A					T_B				
		Матрица					Свободные члены				
$n_1 +$		1	2	3	...	n	$n+1$	$n+2$	...	$n+q$	
1											
2											
3											
...											
n											

		T_B					T_B				
		Матрица					Свободные члены				
$n_1 +$		1	2	3	...	n	$n+1$	$n+2$	...	$n+q$	
1											
2											
3											
...											
n											

		Строка α					Строка α				
		1	2	3	...	n	$n+1$	$n+2$	...	$n+q$	
$n_1 +$											

Рис. 12.

Как известно, в результате такой операции, которая носит название первой (или левой) трансформации Гаусса, матрица становится симметричной и положительно определенной. Трансформация Гаусса находит в приложениях применение не только для решения систем уравнений. В частности, составление системы линейных уравнений метода сил сводится к операции транс-

* Предполагается, что графы n_1 ; $n_1 - 1$; $n_1 - 2 \dots$ и графы $n_1 + v + 1$; $n_1 + v + 2 \dots$ могут присутствовать в таблице, но они не участвуют в операциях ($v = n + q$).

формации матрицы усилий, подобной трансформации Гаусса [55]. О замечательных свойствах симметричных матриц положительной определенной формы и об использовании этих свойств при решении систем линейных уравнений можно прочесть в капитальной работе [67].

Благодаря симметрии получаем возможность экономить ячейки памяти, записывая только половину матрицы B и свободные члены.

Для того, чтобы не загромождать ячейки памяти данными исходной табл. A , будем вводить их в память по одной строке, принимая следующий порядок вычислений:

а) табл. B в памяти очищается от чисел — во все ячейки засылаются нули;

б) в некоторую строку α памяти вводится строка λ табл. A , для чего отводится необходимое количество ячеек. Начальное значение λ равно единице, конечное m ;

в) выполняется программа вычислений для табл. A , состоящей из одной строки α . Результаты суммируются с предыдущими содержимыми ячеек табл. B и засылаются в те же ячейки.

Циклы вычислений (пункты б и в) выполняются для значений $\lambda = 1, 2 \dots m$ (m — количество строк табл. A).

Ввод перфокарт для отдельных строк матрицы выполняется программно, т. е. автоматически.

Для составления адресной программы обозначим фиксаторы: p_1 — фиксатор строки α (p_1 равно адресу, предшествующему первому адресу строки α);

B — фиксатор табл. B (B равно адресу, предшествующему первому адресу табл. B);

D_0 — фиксатор диагонального элемента табл. B (D_0 равно адресу текущего диагонального элемента).

Вначале в фиксатор D_0 засылается адрес первого диагонального элемента $B+1 \rightarrow D_0$ (или $B+1 = D_0$).

Для выявления закономерностей образования адресов диагональных элементов необходимо сначала условиться, в каком порядке нумеровать ячейки таблицы. Так как память машины линейная, а не матричная, то таблица как бы вытягивается в одну линию с непрерывным возрастанием номеров ячеек. Нумерация ячеек по графам и по строкам показана на рис. 13 и 14 для случая, когда $n=5$ и $q=2$. Закономерность образования адресов зависит от способа нумерации ячеек.

Для изучаемой задачи примем нумерацию по строкам. Нетрудно убедиться, что между последующим и предыдущим адресами диагональных элементов существует простая зависимость:

$$D_{\text{послед}} = D_{\text{пред}} + h_1,$$

h_1 — переменный шаг переадресации.

Начальное его значение равно $'n + 'q$; при переходе к каждой следующей строке он уменьшается на единицу.

Итак, для получения последующего адреса диагонального элемента нужно к предыдущему адресу ($'D_0$) прибавить переменный шаг переадресации $'h_1$.

Например, для рис. 14 при $'B=0$, $'n=5$ и $'q=2$ получим последовательные значения для $'D_0$.

Начальные значения для первой строки $'D_0=1$; $'h_1=5+2=7$.

1	2	4	7	11	16	21
	3	5	8	12	17	22
		6	9	13	18	23
			10	14	19	24
				15	20	25

Рис. 13.

$\bar{T}_3$

1	2	3	4	5	6	7
	8	9	10	11	12	13
		14	15	16	17	18
			19	20	21	22
				23	24	25

Строка α

26	27	28	29	30	31	32
----	----	----	----	----	----	----

Строка α в числах

$\lambda=1$	1,0	2,0	1,0	3,0	1,0	2,0	4,0
$\lambda=2$	3,0	1,0	4,0	2,0	1,0	3,0	2,0

Рис. 14.

Последующие значения:

для второй строки $'D_0=1+7=8$, $'h_1=7-1=6$;

для третьей строки $'D_0=8+6=14$, $'h_1=6-1=5$;

для четвертой строки $'D_0=14+5=19$, $'h_1=5-1=4$ и т. д.

Шаг переадресации в строках i и строке α табл. В равен единице.

Из изложенного видно, что при переводе эскизных программ на язык адресного программирования весьма важно установить закономерность образования адресов элементов, участвующих в операциях. Если такие зависимости установлены, то адресная программа по эскизной составляется достаточно просто.

Для данной задачи необходимо определить еще один параметр.

c —количество ячеек в матрице B , включая свободные члены.

Нетрудно убедиться, что $c = \frac{n(n+1)}{2} + n \times q$. Например, при $n=5$ и $q=2$ получим

$$c = \frac{5(5+1)}{2} + 5 \times 2 = 25.$$

Количество ячеек матрицы поместим в ячейку, которую по принятому правилу также назовем c (c —количество ячеек).

Программа первой трансформации Гаусса для матрицы A при q графах свободных членов следующая:

- 1) $0 \rightarrow 'B+1$ до $'B+'c$; (очистка матрицы B);
- 2) $1 \rightarrow \lambda$;
- 3) $'n+'q \rightarrow h_1$;
- 4) $\Pi \rightarrow$ (от $'n_1+1$ до $'n_1+'h_1$); (ввод перфокарт Π с числами строки λ матрицы A);
- 5) $'B+1 \rightarrow D_0$;
- 6) $1 \rightarrow n_2$;
- 7) $0 \rightarrow n_3$;
- 8) $'n_1+'n_2 \rightarrow r$;
- 9) $'(D_0+'n_3) + "r \times ('r+'n_3) \rightarrow ('D_0+'n_3)$;
- 10) $'n_3+1 \rightarrow n_3$;
- 11) $P \{ 'n_3 \leq 'h_1-1 \}$ 9, 12 (или $P \{ 'n_3 = 'h_1 \}$ 12, 9);
- 12) $'n_2+1 \rightarrow n_2$;
- 13) $P \{ 'n_2 \leq 'n \}$ 14, 16;
- 14) $'D_0+'h_1 \rightarrow D_0$;
- 15) $'h_1-1 \rightarrow h_1$ 7;
- 16) $'\lambda+1 \rightarrow \lambda$;
- 17) $P \{ '\lambda \leq 'm \}$ 3, ост.

Отладка адресной программы

Правильность составленной программы проверяют на небольшом примере, выполняя на бумаге все предусмотренные программой адресные операции. В этом состоит так называемая отладка адресной программы. Выполним отладку на примере табл. B , помещенной на рис. 14. Отладка поможет проследить логическую схему счета и выяснить вопросы, если они возникли у читателя.

Параметры задачи: $'n=5$; $'m=5$; $'q=2$.

Для простоты выкладок принимаем $'B=0$, $'n_1=25$.

- 1) $0 \rightarrow$ от 1 до 25;
- 2) $1 \rightarrow \lambda$;
- 3) $5+2=7 \rightarrow h_1$;
- 4) $\Pi \rightarrow$ (от 26 до 32);
- 5) $1 \rightarrow D_0$;
- 6) $1 \rightarrow n_2$;
- 7) $0 \rightarrow n_3$;
- 8) $25+1=26 \rightarrow r$; (или $26 = 'r$);
- 9) $'1+'26 \times '26 \rightarrow 1$;
- 10) $0+1=1 \rightarrow n_3$;
- 11) $P \{ 1 \leq 7-1 \}$ 9*, 12;

* Так как условие выполняется, то управление передается формуле 9.

- 9) $'2 + '26 \times '27 \rightarrow 2$;
 10) $1 + 1 = 2 \rightarrow n_3$;
 11) $P \{2 \leq 6\} \ 9, \ 12$;
 9) $'3 + '26 \times '28 \rightarrow 3$;
 10) $2 + 1 = 3 \rightarrow n_3$.

Далее заполняются ячейки 4, 5, 6 и 7, адреса которых соответствуют значениям $'n_3 = 3, 4, 5, 6$.

- 10) $6 + 1 = 7 \rightarrow n_3$;
 11) $P \{7 \leq 6\} \ 9^*, \ 12$;
 12) $1 + 1 = 2 \rightarrow n_2$;
 13) $P \{2 \leq 5\} \ 14, \ 16$.

Дальше процесс отладки выполняется аналогично до исчерпания программы (до останова).

λ	$i \ (2-5)$				
n_2	1	2	3	4	5
D_0	1	8	14	19	23
h_1	7	6	5	4	3
n_3	0, 1, 2, 3, 4, 5, 6	0, 1, 2, 3, 4, 5	0, 1, 2, 3, 4	0, 1, 2, 3	0, 1, 2

Рис. 15.

При отладке программы незачем выписывать арифметические действия — прибавление единиц (или $'h_1$) к содержимому счетчиков, проверочные формулы и т. п. Это сделано лишь для пояснения.

Обычно результаты отладки выписываются справа на полях программы, а для того, чтобы знать последнее содержимое счетчиков, их величины по мере вычисления заносятся в специальную табличку (рис. 15). После отладки логической схемы счета выполняется счет по полученным адресным формулам для какого-нибудь небольшого численного примера и результаты сравниваются с решением, полученным без применения программы.

Пусть, например, содержимое ячеек строки α будет таким, как показано на рис. 14 внизу.

Выпишем из отладочного примера последовательные формулы счета 9 и подставим в них конкретные числа. Вначале в табл. В написаны нули.

Адреса будем изображать целыми цифрами без запятых, числа — десятичными числами:

* Так как условие не выполняется, то управление передается формуле 12.

$$\begin{aligned}
'1 + '26 \times '26 &= 0,0 + 1,0 \times 1,0 = 1,0 \rightarrow 1; \\
'2 + '26 \times '27 &= 0,0 + 1,0 \times 2,0 = 2,0 \rightarrow 2; \\
'3 + '26 \times '28 &= 0,0 + 1,0 \times 1,0 = 1,0 \rightarrow 3; \\
'4 + '26 \times '29 &= 0,0 + 1,0 \times 3,0 = 3,0 \rightarrow 4; \\
'5 + '26 \times '30 &= 0,0 + 1,0 \times 1,0 = 1,0 \rightarrow 5; \\
'6 + '26 \times '31 &= 0,0 + 1,0 \times 2,0 = 2,0 \rightarrow 6; \\
'7 + '26 \times '32 &= 0,0 + 1,0 \times 4,0 = 4,0 \rightarrow 7.
\end{aligned}$$

Далее вычисления продолжают аналогично и заполняются остальные ячейки 8—25 таблицы. На этом заканчивается первый цикл расчета для $\lambda=1$. Цикл вычислений для $\lambda=2$ выполняется в той же последовательности, но исходными данными служат числа второй строки и числа матрицы B , полученные в предыдущем цикле.

В рассмотренных выше программах пронумерованы все формулы. Хотя можно было нумеровать только те из них, которым передается управление, но сквозная нумерация удобна при переводе адресной программы и составлении рабочей программы для ЭВМ (см. гл. X).

При подготовке адресной программы для программирующих программ института кибернетики АН УССР для циклических процессов применяются сокращенные обозначения:

$$\text{Ц}_N \{N(\text{Ш})K \rightarrow c\} M_u$$

— в счетчик c засылается начальное значение N и ряд последующих значений с шагом переадресации Ш до конечного значения K . До выхода из цикла управление передается следующей по порядку формуле.

После выхода из цикла управление передается формуле, помеченной M_u (M_u — так называемая метка формулы u).

N — номер данного цикла в программе.

Например, простейшая программа, помещенная на стр. 38, сокращенно записывается так ($N=0$; $\text{Ш}=1$; $K='v+1$; $c=m$; $u=1$):

$$0 \rightarrow r;$$

$$\text{Ц}_1 \{0(1)K \rightarrow m\} M_1;$$

$$'r + ('n_1 + 'm) \times ('n_2 + 'm) \rightarrow r;$$

$$M_1 : \text{ост или } M_1 : P_0 M_2.$$

Безусловный переход обозначается символом P_0 . Выражение P_0M означает безусловный переход на изменение параметра в цикле с меткой M_2 .

Достоинством краткой записи является лучшая обозримость программы, что облегчает ее анализ. Однако для перевода программы на язык ЭВМ, если он выполняется программистом, а не машиной, развернутая запись более удобна.

Сокращенная запись адресной программы облегчает кодировку при автоматизации программирования, но для этого в программирующей программе должны быть учтены особенности данной формы записи.

В больших программах нередко используются заранее заготовленные подпрограммы. На адресном языке обращение к подпрограммам выражается формулами вхождения

$$\Phi_1 \rightarrow A, \quad \Phi_2 \rightarrow B, \quad \Phi_3 \rightarrow C, \dots$$

$$П(A, B, C \dots) П_N,$$

где $П$ —знак вхождения в подпрограмму;

A, B, C —фиксаторы параметров задачи;

$П_N$ — N -ая подпрограмма (N —номер, наименование).

Обращению к подпрограмме предшествует засылка в фиксаторы некоторых значений, символически обозначенных Φ .

Для сложных разветвляющихся процессов применяются так называемые признаки, в зависимости от значения которых управление передается тому или иному блоку программы. Признаки засылаются в специальные рабочие ячейки. Применение признаков будет показано в гл. III, IX на примерах более сложных программ решения систем линейных уравнений, расчета сооружений и т. п.

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ АЛГОЛ-60

Язык адресного программирования, хотя и абстрагируется от конкретной машины, все же весьма близок к машинному в том смысле, что структура адресной программы по существу совпадает со структурой рабочей программы. Нужно лишь перевести адресные формулы на язык конкретной ЭВМ, представить их в кодах машины. Это удобно, если переводит человек, но это же приводит к некоторой ограниченности языка.

При автоматизации программирования информация для программирующей программы может в принципе задаваться в более общей и отвлеченной форме, но содержащей все необходимые сведения о рассматриваемом объекте. В идеале можно представить себе такой язык (и основанные на нем программирующие программы), который позволял бы вводить непосред-

ственно в машину математические формулы, алгоритмы и даже словесные указания.

Такой свободой в значительной мере отличается международный алгоритмический язык АЛГОЛ-60, который приближается к формульному и в известной мере словесному тексту (наряду с математическими символами применяются служебные слова).

Это уже не просто система обозначений, а язык с развитой грамматической структурой, со своим синтаксисом (системой правил построения единиц речи), семантикой (трактующей значения слов и оборотов речи) и т. д.

Создание такого всеобъемлющего алгоритмического языка было нелегким делом и потребовало целеустремленных усилий большой группы ученых. Основные положения языка, разработанные Петером Науром (редактором «АЛГОЛ бюллетеня»), обсуждались на международных конференциях 1958—1960 гг. в Копенгагене и Париже. Работа конференций завершилась принятием соглашений о языке АЛГОЛ-60. Сообщение об этом языке было опубликовано на русском языке в 1961 г. в «Журнале вычислительной математики и математической физики» [1].

В настоящее время работа над усовершенствованием языка продолжается [6, 20].

Очень важно, чтобы язык точно передавал смысл и исключалась возможность двоякого толкования одного и того же выражения. Это совершенно необходимо для создания безошибочно действующих программирующих программ (трансляторов), базирующихся на этом языке.

По соглашению на конференциях были приняты три уровня языка:

а) эталонный или определяющий язык, являющийся образцом для всех конкретных представлений, основой и руководством для создания программирующих программ;

б) язык публикаций, допускающий изменения эталонного языка, связанные с удобством печати или написания, применяемый для взаимного обмена информацией;

в) конкретные представления языка, каждое из которых использует совокупность символов конкретной машины, является входным языком программирующей программы для данной машины и сопровождается правилами для перевода с языка публикаций или эталонного языка.

В настоящей работе принят язык публикаций в соответствии с работой [33] с внесением следующих изменений в эталонный язык:

а) латинский алфавит дополнен русскими и греческими буквами;

б) служебные слова даны на русском языке.

Служебные слова на английском языке и соответствующие русские слова:

На английском языке	На русском языке	На английском языке	На русском языке
true	истина	comment	примечание
false	ложь	begin	начало или
go to	на	end	конец или
if	если	boolean	логический
then	то	integer	целый
else	иначе	array	массив
for	для	procedure	процедура
do	цикл	value	значение
step	шаг	while	пока
until	до	real	действительный

В алголе нет понятия адреса и других понятий — счетчиков, фиксаторов и т. п., напоминающих о том, что программа составляется для электронной вычислительной машины.

Вместо адресов служат идентификаторы, которые применяются для обозначений переменных массивов, меток и процедур подпрограмм (обособленных частей программы).

Операция засылки — основная операция адресного (в сущности и рабочего) программирования

$$a \rightarrow b$$

— заменяется в алголе операцией присваивания

$$b := a.$$

Некоторой переменной, обозначенной идентификатором b , присваивается значение a . Знак присваивания $:=$ (двоеточие равно).

Содержание обеих формул одинаково. Для реализации операции присваивания в ЭВМ необходимо для хранения переменной b и ее значения a предусмотреть некоторые ячейки. Если обозначить адреса этих ячеек теми же символами a и b , то получим приведенную выше адресную запись операции присваивания.

Особенность алгола состоит в том, что не нужно заботиться о размещении информации (даже в символическом виде) в ячейках памяти машины. Эта задача решается программирующей программой. Поэтому отпадает необходимость в адресах.

Тем не менее операция формирования адресов все же присутствует в алголе. Для переменных величин, которые изображают собой не реальные, а отвлеченные числа (например, адреса), используется класс целых чисел, обозначаемых набранным жирно служебным словом **целые** (по английски **integer**) в отличие от действительных чисел, обозначаемых словом **действительные** (**real**). Имеется еще класс чисел, которые могут прини-

мать только два значения — истина и ложь — и обозначаются словом логические (или булевские — *boolean*).

Десятичные цифры изображаются как обычно (в эталонном языке вместо запятой ставится точка).

В отличие от адресного программирования, в котором адреса, как и в рабочих программах, располагаются в линейной последовательности, в алголе возможно изображать не только одномерные, но и многомерные массивы чисел. Границы массивов чисел указываются в индексных скобках, перед которыми ставится идентификатор (обозначение) массива. В этом отношении обозначения на алголе аналогичны обозначениям в инженерных программах (хотя эти системы имеют разные назначения и создавались независимо).

№ п.п	Обозначения в инженерных программах	Обозначения на алголе	Содержание
1	$\begin{array}{ l} \text{стр } i \text{ } ^n \text{ } T_N \\ \text{гр } i \text{ } ^n \text{ } T_N \end{array}$	$TV[1:n]$	Линейный массив (строка, графа) в пределах от 1 до n
2	$T_N \begin{array}{ l} 1, n \\ 1, m \end{array}$	$TN[1:m, 1:n]$	Двухмерный массив (таблица) в пределах от 1 до m и от 1 до n
3	$i. kT_N$	$TN[i, k]$	Ячейка на пересечении строки i и графы k . i, k — координаты ячейки в массиве

В индексных скобках указывается список граничных пар, определяющий пределы изменения адресов. Одна пара $(1:n)$ соответствует линейному массиву, две пары $(1:m, 1:n)$ — двумерному массиву, то есть матрице. Имеется возможность изображения пространства трех (три граничных пары) и большего числа измерений.

Интересно отметить, что в инженерном программировании применяются и некоторые служебные слова (стр — строка, гр — графа, Σ — сумма, проверка), которые необходимы для обозначения специфических операций на электрических автоматах — суммирования произведений или частных, построчных проверок. При программировании для ЭВМ эти слова не нужны.

Обозначение таблицы или части ее заменяется идентификатором массива — сочетанием букв и цифр, причем первый символ обязательно должен быть буквой. Поэтому вместо $T_1, T_2 \dots T_N$ на алголе пишется $T1, T2, TN$ (цифры и буквы на одном уровне).

С помощью индексных скобок можно изображать адреса высоких рангов. Например, адресная формула

$$'('A+'i+'k)$$

на алголе имеет вид

$$A[i+k],$$

где i и k —некоторые переменные величины, определяющие положение ячейки в линейном массиве A .

Если переменная k принадлежит к некоторому другому массиву B , то получим

$$A[i+B[k]].$$

Последнее выражение соответствует адресу третьего ранга и в адресном виде запишется так:

$$'('A+'i+'('B+'k)).$$

Запись $A[B[i], k]$ определяет положение ячейки в массиве A , причем одна из переменных i принадлежит другому массиву B . По существу этим определяется положение ячейки в трехмерном массиве (рис. 16). Аналогично можно реализовать и адреса более высоких рангов, например:

$$A[i+B[k, c[v]]+D[k+E[v]]].$$

В отличие от алгебраических формул сложные выражения на алголе изображаются многократным применением скобок одного вида.

Квадратные скобки применяются в индексных выражениях. В арифметических выражениях применяются круглые скобки. Сложное арифметическое выражение с применением скобок имеет, например, следующий вид:

$$(a+(b+c \times (d+l))/f) \times g.$$

Для обозначения алгебраических действий применяются следующие символы, записанные в порядке возрастания старшинства:

$+$, $-$ —сложение, вычитание;
 $\times$, $/$, $\div$ —умножение, деление, целочисленное деление;
 $\uparrow$ —возведение в степень.

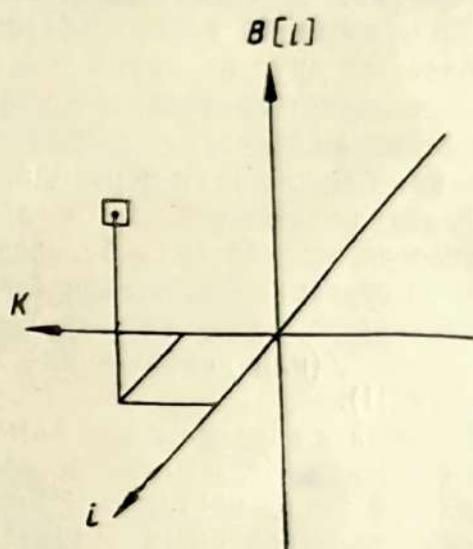


Рис. 16.

Старшинство действий и правило раскрытия скобок такое же, как в алгебре. Например, приведенное выше сложное арифметическое выражение раскрывается так:

$$d+l=d1; c \times d1=c1; b+c1=b1;$$

$$b1/f=f1; a+f1=a1; a1 \times g.$$

Количество знаков раскрытия и закрытия скобок (,) должно быть одинаковым, что используется для контроля записи.

Большим достоинством алгола является то, что он позволяет простыми средствами (например, расстановкой скобок) осуществлять глубокие построения. Так же, как и в алгебре, любое арифметическое выражение, сколько угодно сложное, но заключенное в скобки, может рассматриваться в качестве первичного выражения, то есть как одна величина.

Программа на алголе состоит из ряда формул-операторов, отделяемых друг от друга точкой с запятой и выполняемых в той последовательности, в которой они записаны. Для изменения порядка следования формул используются операторы условного или безусловного перехода.

В алголе так же, как и в адресном языке, применяются различные логические (или булевские) выражения, к которым относятся операции отношения ($=$, $\neq$, $>$, $\geq$, $<$, $\leq$) и операции логического умножения $\wedge$ (или конъюнкции), логического сложения $\vee$ (или дизъюнкции), импликации, $\supset$, отрицания $\neg$ ([26], гл. II).

Но если в адресном программировании проверка изображается символами $P\{\text{логическое условие}\}$ u, v , то та же проверка на алголе записывается с помощью служебных слов **если, то, иначе, на**, выделяемых жирно:

если $\langle \text{логическое условие} \rangle$ **то** u **иначе** v .

Если логическое условие удовлетворяется (принимает значение «истина»), то переходим на формулу u , иначе (при значении ложно) на формулу v .

u и v — метки (номера) формул.

Выражение, указывающее, на какую формулу следует перейти, называется именующим.

Возможны и другие написания проверок без слова **иначе**.

Например, **если** $\langle \text{логическое условие} \rangle$ **то** u ;

Если логическое условие удовлетворяется, переходим на формулу u , в противном случае на следующую по порядку формулу (следующую после точки с запятой);

если $\langle \text{логическое условие} \rangle$ **то** $h1 := h1 + 1$;

Если логическое условие удовлетворяется, то выполняется оператор присваивания (в рассматриваемом примере переменной $h1$ присваивается значение $h1 + 1$), в противном случае переходим к выполнению следующей по порядку формулы.

Наконец в условном операторе может быть целая цепь **если** и **то**.

Например,

если $B1$ **то** $S1$ **иначе если** $B2$ **то** $S2$
иначе если $B3$ **то** $S3$ **иначе** : $S4$.

Здесь $B1, B2, B3$ —логические выражения; $S1, S2, S3$ — безусловные операторы, которыми могут быть именуемые выражения, операторы присваивания, пустые операторы. После слова **то** должно стоять простое выражение или сложное, которое с помощью скобок превращается в простое. Эффект действия сложного выражения сводится в конечном счете к указанным трем видам операторов.

Пустой оператор (то есть оператор, в котором ничего не записано) может применяться для придания определенного направления процессу вычислений.

Указанная выше цепочка логических условий, представляющая в совокупности полный условный оператор, просматривается слева направо, пока не будет найдено условие, принимающее значение «истина». Далее выполняется следующий за этим условием безусловный оператор. После этого, если в безусловном операторе содержится именуемое выражение, выполняется указанная в нем формула, если же другое выражение (операция присваивания, пустой оператор), выполняется оператор $S4$, следующий за полным условным оператором (за точкой с запятой). Пустой оператор в данном случае может быть использован для прерывания полного условного оператора и перехода к следующей формуле. Для изображения меток, условно обозначенных выше u, v , могут использоваться любые числа, буквы, слова или их сочетания, после которых поставлено двоеточие. Например, выражение $M10 : h1 := h1 + 1$ означает: — метка $M10$: переменной $h1$ присвоить значение $h1 + 1$.

Если после двоеточия ничего не стоит

$M10 :$

то будем иметь пустой оператор.

Начало и конец программы обозначаются служебными словами **начало**, **конец**.

Для обозначения цикла вычислений также применяются служебные слова: **для**, **шаг**, **до**, **цикл**.

Например, обозначение **для** $m := i$ **шаг** 1 **до** n **цикл** $A [D0 + m] := 0$ выражает следующее: для переменной m при начальном значении i , шаге переадресации 1 и конечном значении n выполнить то, что указано после слова **цикл**, в данном примере последовательно присвоить величине $A [D0 + m]$ значение нуль.

После исчерпания цикла (то есть выполнения оператора цикла для последнего значения переменной) выполняется формула, следующая за оператором цикла (за точкой с запятой).

В рассмотренном примере оператор цикла был простым, то есть состоял из одного выражения. В тех случаях, когда оператор цикла является составным и состоит из нескольких выражений (в число которых могут входить и внутренние циклы), для указания его границ применяются операторные скобки, состоящие из служебных слов *начало* (обозначающее открытие скобки) и *конец* (закрытие скобки). Вместо этих слов можно применять фигурные скобки { . . . }, что целесообразно при коротких составных операторах. Таким образом с помощью операторных скобок можно весьма просто изобразить многократное циклирование (циклы в циклах). Цикл, в котором содержится другой цикл, является старшим по отношению к последнему*.

Схему выполнения цикла можно изобразить так [1]: начало; проверка; оператор *S*; продвижение; следующий оператор.

Начало — присваивается первое значение переменной; выполняется оператор цикла *S*.

Продвижение — присваивается следующее значение переменной.

Проверка — проверяется, не исчерпан ли цикл; если цикл не исчерпан, то снова выполняется оператор цикла *S*, в противном случае — выход из цикла и переход к следующему оператору.

Переменная (в рассматриваемом примере *m*), список значений которой указан в заголовке до слова *цикл*, называется параметром цикла.

Если параметр принимает всего два-три значения, а также в тех случаях, когда шаг переадресации изменяется, можно применить такую запись: для $m := 1, 3, 10$ цикл <оператор>, либо для $m := 1, 3$ шаг 1 до 7, 10 цикл <оператор>.

где 1, 3, 10 (1, 3 до 7, 10) — значения, которые последовательно присваиваются переменной *m*.

Цикл может быть описан также следующим образом:

для $m := \langle \text{арифметическое выражение} \rangle$ пока <логическое выражение> цикл <оператор цикла>.

Параметру *m* последовательно присваивается значение, численное в арифметическом выражении, до тех пор, пока соблюдается некоторое логическое условие (логическое выражение имеет значение «истина»), в противном случае (при значении «ложь») — выход из цикла и переход к следующему оператору.

* Здесь можно применить наглядное понятие цикла *n*-го ранга (аналогичное адресу *n*-го ранга). Цикл *n*-го ранга содержит цикл *n*—1-го ранга, который в свою очередь содержит цикл *n*—2-го ранга и т. д.

Например, для $m := c + q$ пока $m > \varepsilon$ цикл <оператор цикла>.

Цикл вычислений выполняется до тех пор, пока m больше некоторой требуемой точности расчета.

В операторе цикла может быть выполнено, например, решение системы линейных уравнений последовательными приближениями. При этом c представляет собой результат подстановки неизвестных в суммарное уравнение, а q —сумма свободных членов.

Иногда возникает необходимость в изменении меток в именуемых выражениях в процессе счета. Для этого применяется так называемый переключатель (switch).

В описании переключателя, которое дается в начале программы, указывается идентификатор переключателя и список меток, которые ему присваиваются. Например,

переключатель $S := A, B, C$.

В самой программе вместо соответствующего именуемого выражения помещается идентификатор, а в индексных скобках переменная — порядковый номер метки в списке.

Например,

на $S[i]; u;$

Величина $S[i]$ при $i=1$ заменяется для рассматриваемого примера первой по порядку меткой A , при $i=2$ —второй меткой B , при $i=3$ —третьей меткой C .

Если $i=4$, а четвертой по порядку метки нет, то выполняется формула u , следующая после точки с запятой.

Ниже для пояснения приведен отрезок программы, в которой использован переключатель. Формулы и описания, не имеющие отношения к переключателю, изображаются многоточиями.

Целые

Действительные массивы

Переключатель $v := 5, 7, 11;$

1: $i := 1$	:	7:	:
2:	:	8:	:
3:	:	9:	:
4:	:	10: $i := 3;$	на 2;
5: $i := 2;$	на 2;	11:	:
6:	:	12:	:

По формуле 1 переменной i присваивается значение единица. Выполняются формулы 2 и 3. По формуле 4 управление передается $v[1]$, т. е. на первую в переключательном списке метку 5. Выполняется формула 5: переменной i присваивается значение 2, управление передается формуле 2. Выполняются формулы 2 и 3.

По формуле 4 управление передается формуле v [2], то есть на вторую в переключенном списке метку 7.

Выполняются формулы 7, 8 и 9.

По формуле 10 переменной i присваивается значение 3, управление передается формуле 2. Выполняются формулы 2 и 3.

По формуле 4 управление передается v [3], то есть третьей в переключательном списке метке 11.

Выполняются 11 и последующие формулы программы.

В тех случаях, когда в программе имеются многократно повторяемые обособленные части (подпрограммы), то для сокращения записи каждая такая часть, называемая процедурой, описывается в начале программы. Описание процедуры состоит из заголовка и тела процедуры.

В заголовке процедуры указывается ее идентификатор (обозначение), список формальных параметров (в скобках), характеризующих процедуру, тип этих параметров и их значение.

Например, заголовок:

Процедура решение уравнений (a, b, c)

массив a ; целые b, c ; значения b, c .

В этом заголовке идентификатором процедуры являются слова «решение уравнений».

За заголовком процедуры помещается ее программа, называемая телом процедуры.

В тексте основной программы (в начале которой дано описание процедур) обращение к выполнению тела процедуры производится с помощью оператора процедуры, в котором указан ее идентификатор и список фактических параметров.

До выполнения процедуры формальным параметрам, указанным в заголовке после слова **значение**, присваиваются значения фактических, а формальные параметры, не указанные в списке значений, заменяются фактическими в самом теле процедуры. Здесь есть существенное различие: в первом случае процедура не меняется, так как операции присваивания предшествуют процедуре, во втором случае формулы процедуры изменяются, так как при ее выполнении вместо формальных фигурируют фактические параметры.

Например, обращение
решение уравнений (A, n, q)

обозначает, что в процедуре формальный параметр a всюду заменяется фактическим параметром A , а формальные параметры b и c остаются на своих местах без изменения, так как значения n и q присваиваются им до выполнения процедуры:

$$b := n; c := q.$$

Если n и q — арифметические выражения, то они вычисляются только один раз — перед выполнением процедуры. Для

параметров, не перечисленных в списке значений, вычисления выполняются каждый раз, когда соответствующие арифметические выражения, заменяющие формальные параметры, встречаются в процедуре.

Если описание параметров процедуры дано в начале основной программы и они при всех обращениях сохраняют свое значение, то формальные параметры в заголовке процедуры и фактические параметры в операторе могут не указываться.

В некоторых случаях приходится прибегать к нестандартным процедурам, которые на алголе не описываются. В этом случае необходимые пояснения даются в разделе **примечание**.

Программа на алголе обычно имеет следующую структуру (см. схему, помещенную ниже):

1) в начале программы описываются применяемые процедуры, если таковые имеются;

2) описываются входящие в программу массивы и переменные величины, то есть указывается, к какому типу они относятся (целые, действительные, логические);

3) далее следуют формулы программы, отделяемые друг от друга точкой с запятой;

4) начало и конец программы ограничивается служебными словами **начало** и **конец**. Если в программу входят составные операторы, то появляются операторные скобки из слов **начало** и **конец**. Рекомендуются каждую последующую пару скобок сдвигать немного вправо относительно предыдущей. Такое расположение «лесенкой» по старшинству циклов облегчает чтение программы;

5) пояснения к программе, если они необходимы, даются после слова **примечание** в любом месте программы. В разделе **примечание** могут содержаться любые знаки препинания, кроме точки с запятой. После окончания примечаний ставится точка с запятой.

Схема программы

```
Начало. Примечание . . . . . ;
Процедура . . . . . ;
целые . . . . . ; действительные . . . . . ;
. . . . . ; логические . . . . . ;
<тело программы>
конец
```

Особенности построения программ на алголе видны на следующих примерах.

Простейшая инженерная программа

$$\begin{array}{ccc} |\text{стр } a \times \text{стр } b|_1^n & \rightarrow & |\text{стр } c|_1^n \\ (\text{гр}) & (\text{гр}) & (\text{гр}) \end{array}$$

В скобках указаны возможные варианты программы с частичной или полной заменой строк на графы.

На алголе данную программу можно изображать в двух формах — развернутой и сокращенной с применением оператора цикла.

Развернутая форма

Начало

целые i, n , действительные массивы $a, b, c [1:n]^*$;

$i := 1$;

$M1: c[i] := a[i] \times b[i]; i := i + 1$;

если $i \leq n$ то на $M1$

конец

Вначале переменной i , изображающей порядковый номер ячейки в массиве, присваивается первое значение, равное единице. Затем умножаются числа $a[i] \times b[i]$ и результат присваивается величине $c[i]$ (или, что то же самое, результат засылается в ячейку $c[i]$).

После этого присваивается новое значение переменной i (увеличенное на единицу, поскольку шаг переадресации равен единице) и проверяется, не исчерпаны ли все предусмотренные программой значения i . Если проверка удовлетворяется ($i \leq n$), то переходим к выполнению формулы с меткой $M1$, то есть продолжаем вычисления. Если проверка не удовлетворяется ($i > n$), цикл исчерпан и вычисления прекращаются, что указано словом конец (либо переходим к выполнению следующего оператора, если программа продолжается дальше).

Сокращенная запись той же программы с применением оператора цикла выглядит так:

Начало

целые i, n ; массивы $a, b, c [1:n]$;

для $i := 1$ шаг 1 до n цикл

$c[i] := a[i] \times b[i]$

конец

* Здесь и далее программе предшествует присвоение величине n определенного значения.

Если в описании массива или переменной не указано, к какому классу они относятся, то это значит, что они имеют тип **действительный**. Поэтому слово **действительный** при описании массивов можно не писать.

Видоизменим приведенную выше инженерную программу, поставив слева знак суммы, справа адрес S ячейки, в которую следует записать полученную сумму произведений

$$\Sigma | \text{стр } a \times \text{стр } b |_1^n \rightarrow S$$

(гр) (гр)

Соответствующая программа на алголе будет:

Начало

целые i, n, S ;

массивы $a, b[1:n]$;

$S := 0$;

для $i := 1$ шаг 1 до n цикл

$S := S + a[i] \times b[i]$

конец

Вначале переменной S присваивается значение нуль. Затем при выполнении оператора цикла для i от 1 до n к предыдущим значениям S последовательно прибавляются произведения чисел $a[i] \times b[i]$ и результаты присваиваются величине S . Таким образом в S накапливается сумма попарных произведений в пределах от 1 до n , то есть реализуется рассмотренная инженерная программа.

Перейдем теперь к более сложной программе — к рассмотренной в предыдущем разделе программе первой трансформации Гаусса.

Вначале не будем заботиться об экономии памяти машины и расположим исходную и трансформированную матрицы в разных и притом прямоугольных массивах A и B (рис. 17).

Тогда инженерная программа запишется так:

для $i = 1$ до n , $k = i$ до $n + q$

$$\Sigma | \text{гр } i \times \text{гр } k |_1^m A \rightarrow i k B \rightarrow k. i B.$$

Матрица A — прямоугольная с размерами $m \times n$; матрица B — квадратная порядка n ; q — количество граф свободных членов.

Так как табл. A и B обозначены буквами, а не цифрами, то обозначение T можно не писать*.

На алголе указанная программа примет следующий вид:
Начало Примечание m и n — размеры исходной прямоугольной матрицы A (m —число строк, n —граф), n —порядок транс-

T_A

	1	2	...	n	$n+1$	$n+2$	...	$n+q$
1								
2								
3								
...								
m								

T_B

	1	2	...	n	$n+1$	$n+2$	...	$n+q$
1								
2								
...								
n								

Рис. 17.

формированной квадратной матрицы B , q —количество граф свободных членов. Перед решением задачи память очищается. В качестве операторных применяем фигурные скобки;
 целые i, k, μ, n, m, q ; массивы $A[1:m, 1:n+q]$, $B[1:n, 1:n+q]$;

для $i := 1$ шаг 1 до n цикл
 для $k := i$ шаг 1 до $n+q$ цикл
 для $\mu := 1$ шаг 1 до m цикл

$$B[i, k] := B[i, k] + A[\mu, i] \times A[\mu, k];$$

если $k \leq n$ то $B[k, i] := B[i, k]$
 конец

Чтобы можно было проследить, как реализуется данная программа, ниже помещена ее отладка.

Первоначальные содержимые ячеек матрицы B —нули.

$$k := i := 1,$$

$$\mu := 1; B[1, 1] := B[1, 1] + A[1, 1] \times A[1, 1],$$

$$\mu := 2; B[1, 1] := B[1, 1] + A[2, 1] \times A[2, 1],$$

* Если таблицы обозначены цифрами, то символ T писать обязательно $T_1, T_2, \dots$ или на алголе $T1, T2, \dots$

$$\mu = m; B[1, 1] := B[1, 1] + A[m, 1] \times A[m, 1];$$

цикл для μ исчерпан

$$B[1, 1] := B[1, 1].$$

Последняя операция приобретает смысл при $i \neq 1$ (см. ниже).

Произведенные вычисления эквивалентны выполнению формулы инженерной программы:

$$\Sigma | \text{gr } 1 \times \text{gr } 1 |_1^m A \rightarrow 1.1 B;$$

Меняем на один шаг переменную k .

$$k := 2;$$

$$\mu := 1; B[1, 2] := B[1, 2] + A[1, 1] \times A[1, 2].$$

$$\mu := 2; B[1, 2] := B[1, 2] + A[2, 1] \times A[2, 2].$$

.....

$$\mu := m; B[1, 2] := B[1, 2] + A[m, 1] \times A[m, 2];$$

$$B[2, 1] := B[1, 2].$$

Произведенные вычисления эквивалентны выполнению формулы инженерной программы:

$$\Sigma | \text{gr } 1 \times \text{gr } 2 |_1^m A = 1.2 B = 2.1 B.$$

Далее переменной k таким же образом последовательно присваиваются значения $3, 4, \dots, n+q$ и вычисляются соответственно значения $B[1, 3], B[1, 4], \dots, B[1, n+q]$ до исчерпания цикла.

Теперь изменяется на один шаг значение i , а величине k присваивается согласно программе начальное значение, равное i .

$k := i := 2$, и повторяются циклы для μ и k до их исчерпания.

Снова изменяем на один шаг значения i , а k присваивается начальное значение, равное i .

$$k := i := 3 \text{ и т. д.}$$

Аналогично вычисления продолжаются до исчерпания старшего цикла для i .

Представление матричных операций на языке АЛГОЛ-60 очень удобно при использовании двумерных массивов. К сожалению, при этом во многих случаях уменьшается возможность экономии оперативной памяти машины. Например, в рассмотренной программе не учитывается, что ввиду симметрии матрицы элементы, расположенные ниже главной диагонали, можно не записывать и не занимать ими ячейки памяти. Кроме того, как

было указано в предыдущем разделе, матрицу A целесообразно вводить в память машины не всю сразу, а построчно на одно и то же место и выполнять трансформацию для табл. A , состоящей из одной строки.

Применение этих и других приемов позволяет в несколько раз уменьшить объем ячеек памяти, необходимый для решения задачи.

Однако для непрямоугольных матриц (треугольных и более сложных форм) нельзя представлять таблицу в виде двумерного массива. Приходится разворачивать таблицу, как это делалось при адресном программировании, в линейную последовательность ячеек, то есть пользоваться одномерным (линейным) массивом (см. рис. 14).

Ниже помещена программа трансформации Гаусса на алгоритме, при составлении которой приняты те же предпосылки, что и в адресной программе, приведенной в предыдущем разделе.

Начало примечание m и n — размеры прямоугольной матрицы A , n — порядок матрицы B , q — количество граф свободных членов. Элементы, расположенные ниже главной диагонали матрицы B , в память не записываются.

$v = 0,5n \times (n+1) + n \times q$ — количество элементов в матрице B и q графах свободных членов.

$B[1:v]$ — матрица и свободные члены, рассматриваемые как линейный массив.

Программа предусматривает введение в рабочую строку памяти $a[1:n+q]$ последовательно по одной строке матрицы A с выполнением для нее как для самостоятельной матрицы трансформации с накоплением результатов в матрице B .

i и k — номера элементов строки a ,

D — номер диагонального элемента матрицы B ,

j — счетчик вводов строк матрицы A .

Чтение $(a, a1, a2)$ — нестандартная процедура ввода строки матрицы A в ячейки от $a[a1]$ до $a[a2]$. В качестве операторных скобок используем слова **начало**, **конец**;

целые n, q, D, i, k, j ; массивы $a[1:n+q], B[1:v]$;

для $j := 1$ шаг 1 до m цикл

начало $D1 := 1$; чтение $(a, 1, n+q)$;

для $i := 1$ шаг 1 до n цикл

начало

для $k := i$ шаг 1 до $n+q$ цикл

$B[D+k-i] := B[D+k-i] + a[i] \times a[k]$;

$D := D + n + q - i + 1$

конец i

конец j

конец

Отладка данной программы на небольшом примере (см. рис. 18) при $m=n=4$ и $q=2$ выполнена ниже.

Отладка программы

j	D	i	k	Формула
1 (2, 3, 4)	1	1	1	$B[1] := B[1] + a[1] \times a[1]$
			2	$B[2] := B[2] + a[1] \times a[2]$
			3	$B[3] := B[3] + a[1] \times a[3]$
			4	$B[4] := B[4] + a[1] \times a[4]$
			5	$B[5] := B[5] + a[1] \times a[5]$
			6	$B[6] := B[6] + a[1] \times a[6]$
	7	2	2	$B[7] := B[7] + a[2] \times a[2]$
			3	$B[8] := B[8] + a[2] \times a[3]$
			4	$B[9] := B[9] + a[2] \times a[4]$
			5	$B[10] := B[10] + a[2] \times a[5]$
			6	$B[11] := B[11] + a[2] \times a[6]$
	12	3	3	$B[12] := B[12] + a[3] \times a[3]$
			4	$B[13] := B[13] + a[3] \times a[4]$
			5	$B[14] := B[14] + a[3] \times a[5]$
			6	$B[15] := B[15] + a[3] \times a[6]$
	16	4	4	$B[16] := B[16] + a[4] \times a[4]$
			5	$B[17] := B[17] + a[4] \times a[5]$
			6	$B[18] := B[18] + a[4] \times a[6]$

Данная программа незначительно сложнее предыдущей, составленной для двумерного массива (вычисляются номера диагональных и других элементов), но дает значительную экономию памяти. Если для предыдущей программы при размещении табл. А и В требуется $n(m+n+2q)$ ячеек памяти, то для данной программы — всего $n(0,5n+1,5+q)+q$ ячеек. Например, при $m=80$, $n=40$ и $q=10$ получим для первой программы $40(80+40+20)=5600$, а для второй программы $40(20+1,5+10)+10=1270$ ячеек.

В	1	2	3	4	5	6
		7	8	9	10	11
			12	13	14	15
				16	17	18
α	1	2	3	4	5	6

Рис. 18.

А				С			
	1	2	3		1	2	3
1	1.0	1.0	2.0	1	1.0	3.0	2.0
2	2.0	3.0	3.0	2	2.0	3.0	4.0
3	2.0	4.0	3.0	3	3.0	2.0	1.0
Д				В		E*	
	1	2	3	1		1	3
1	3	1	2	1	2	1	3
2	1	3	1	2	3	2	1
3	2	1	1	3	2	3	2

Рис. 19.

Таким образом для этого примера при использовании машин с памятью 2048—4096 ячеек вторая программа является единственно возможной.

Приведем отрезок, взятый из реальной программы, в котором используется полный условный оператор из сочетания нескольких если, то, иначе.

для $j:=0, 1, 2$ цикл

для $k:=j \times n + 1$ шаг 1 до $j \times n + n$ цикл

для $l:=1$ шаг 1 до n цикл

для $d:=1$ шаг n до $l+2 \times n$ цикл

для $i:=1$ шаг 1 до m цикл

(если $j=0$ то $h:=\Delta z[i] + \Delta w[i]$ иначе

если $j=1$ то $h:=\cos \theta[i]$ иначе $h:=\sin \theta[i]$;

$T3[k, d] := T3[k, d] + T2[i, k] \times T1[i, l] \times h$).

Здесь Δz , Δw , $\cos \theta$, $\sin \theta$ идентификаторы массивов, обозначения которых совпадают с наименованиями функций в матема-

тических формулах. Этот прием облегчает проверку соответствия формул на алголе исходным формулам.

В заключение рассмотрим пример программы, в которой применяются адреса высоких рангов.

Начало целые i, k, v ; целые массивы $B, E[1:n], D[1:m, 1:n]$; действительные массивы $A, C[1:m, 1:n]$;

для $i := 1$ шаг 1 до n цикл

для $k := i$ шаг 1 до n цикл

для $v := 1$ шаг 1 до n цикл

$$A[B[i], k] := A[B[i], k] + C[B[i], D[k, E[v]]]$$

конец

Последовательность раскрытия индексных скобок видна на простейшем примере для $n=3$ (рис. 19).

При $i=1, k=1, v=1$ получим

$$B[1]=2, E[1]=3, D[1, E[1]]=D[1, 3]=2,$$

и формула примет вид

$$A[B[1], 1] := A[B[1], 1] + C[B[1], D[1, E[1]]]$$

или

$$A[2, 1] := A[2, 1] + C[2, 2] = 2,0 + 3,0 = 5,0.$$

При других значениях переменных вычисления выполняются аналогично.

СРАВНЕНИЕ СИСТЕМ ПРОГРАММИРОВАНИЯ И ПРЕДЛОЖЕНИЯ ПО ИХ УЛУЧШЕНИЮ

Итак, были рассмотрены три системы программирования: инженерное, адресное и на языке АЛГОЛ-60. Первая из этих систем предназначается непосредственно для малой механизации — электрических автоматов и является одновременно эскизом (заданием) для двух других систем, используемых при составлении обобщенных программ для электронных цифровых машин.

Очевидно, программа на алголе может иметь то же строение, что и адресная программа. Нетрудно переписать любую адресную программу в терминах алгола. Для этого достаточно заменить операцию засылки операцией присваивания, поменяв местами левую и правую части адресной формулы, а проверку логического условия $P\{ \dots \}$ и, v заменить конструкцией **если** $\dots$ **то на** i **иначе на** v . По существу это даже не перевод с одного языка на другой, который предполагает перефразировку, связанную с различным строением языков, а именно переписи-

сывание лишь с частичным изменением терминов, так как арифметические и логические операции на обоих языках изображаются одинаково.

И все же это не один и тот же язык.

Адресный язык можно рассматривать как частный случай алгола, но последний значительно шире первого и позволяет осуществлять эффективные построения, недоступные адресному языку. Поэтому перевод с алгола на адресный язык не всегда так же прост, как обратный процесс.

Кардинальное отличие алгола состоит в том, что он может оперировать не только линейными, но и многомерными массивами; кроме того, в одной формуле алгола может присутствовать целый ряд арифметических и логических выражений (а не одно выражение того или другого рода), что позволяет формулировать программы наиболее сжато.

Приведем два отрезка программы на алголе.
Первый вариант:

M5: если $h \leq i + m1 + 1$ то на M6; $j1 := j1 + 1$;

M6: если $i + m1 \leq n - h$ то на M7; $j2 := j2 + 1$;

M7:

конец $m1$.

Пустой оператор M7 служит для передачи управления на конец цикла для $m1$ с последующим продолжением цикла до его исчерпания.

Здесь сохранена конструкция, характерная для адресного программирования.

Тот же отрезок можно записать короче.

Второй вариант:

M5: если $h > i + m1 + 1$ то $j1 := j1 + 1$;

если $i + m1 > n - h$ то $j2 := j2 + 1$;

конец $m1$.

Второй вариант больше соответствует возможностям алгола и позволяет исключить ряд именуемых выражений и меток.

АЛГОЛ-60 является международным языком, что дает возможность организовать обмен программами между научными коллективами разных стран и создать большие библиотеки стандартных программ.

Создать всеобъемлющие программирующие программы для алгола нелегко, и для этого потребуется время. Известно, что программирующие программы, уже созданные для некоторых машин, состоят из десятков тысяч команд и размещаются на лентах.

Но это не значит, что пока не будут созданы и отлажены безукоризненные программирующие программы для алгола, он не имеет практического смысла. Алгол можно и нужно использовать (как и адресный язык) уже сейчас для формулировки обобщенных программ, и если программирующей программы еще нет, то последующий перевод на язык конкретной ЭВМ может быть выполнен программистом. В последнем случае целесообразно составлять на алголе развернутые программы без каких-либо сокращений (в том числе и без сокращения записи циклов), сохраняя адресную структуру и нумеруя все формулы. Идентификатор массива заменяется его фиксатором и прибавляется к индексному выражению в квадратных скобках (как в адресных программах).

При переводе и записи рабочей программы на бланки слева записывается номер формулы на алголе, а справа — одна или несколько команд, соответствующих этой формуле. Такой способ записи облегчает не только технику перевода, но и его проверку другим лицом, что весьма важно.

Как указывалось, алгол по существу охватывает и инженерное программирование, предназначенное для малой механизации — электрических автоматов.

В инженерном программировании имеется ряд символов, позволяющих формулировать программы более кратко, чем на языке АЛГОЛ-60: знак Σ , обозначения строк и граф, объединение одними пределами произвольного количества выражений и т. д.

Однако в инженерных программах слабо развито описание логических операций. Слова **проверка**, указывающего продолжать ли счет или повторить предыдущий этап вычислений, недостаточно для описаний более сложных логических действий, связанных с передачей управления любому, произвольному блоку программы.

Для придания большей определенности в язык инженерного программирования предлагается ввести некоторые элементы алгола:

1. Формулы в программах разделяются точкой с запятой.

2. Для указания о переходе к той или иной части программы применяется одно из выражений

если <логическое выражение> **то на** *и* **иначе на** *у*,
либо

если <логическое выражение> **то на** *и*.

Во втором выражении при несоблюдении логического условия управление передается следующей по порядку формуле.

и и *у* —метки (номера) формул.

3. Формула, на которую передается управление, снабжается меткой, после которой ставится двоеточие.

4. Циклы изменения параметров описываются с помощью заголовка.

Например,
для $i=1$ до m цикл
для $k=n$ до 1 цикл
для $v=1$ до q цикл и т. д.

Слово шаг можно опустить в тех случаях, когда он равен ± 1 .

Порядок написания параметров указывает, что каждому значению предыдущего параметра соответствует полный цикл изменения последующего параметра. Например, приведенное выражение расшифровывается так:

$i=1, k=n, v=1$ до q ;
 $i=1, k=n-1, v=1$ до q ;
 $i=1, k=n-2, v=1$ до q ;
.....
 $i=1, k=1, v=1$ до q ;

 $i=2, k=n, v=1$ до q ;
 $i=2, k=n-1, v=1$ до q ;
 $i=2, k=n-2, v=1$ до q ;
.....
 $i=2, k=1, v=1$ до q ;
и так далее до $i=m$.

5. Вводятся операторные скобки в виде слов **начало**, **конец** либо в виде фигурных скобок. Круглые скобки и их сочетания имеют то же значение, что и в АЛГОЛ-60.

В написании инженерных программ допускается относительная свобода: можно вводить новые слова, некоторые слова не писать и т. п.

В левой части табл. 1 приведен ряд инженерных программ, дополненных указанными элементами алгола.

Если внести в эти программы ряд формальных изменений, соответствующих алголу (идентификатор массива ставить впереди, пределы указывать в индексных скобках), то получим запись, представленную в правой части табл. 1, которую можно назвать сокращенным алголом.

Обе приведенные формы записи можно использовать для изложения алгоритмов, предназначенных для электрических автоматов. Сопоставляя эти формы, убеждаемся, что запись слева удобнее, так как она позволяет объединять одними пределами и обозначениями любое количество выражений.

Сокращенную форму алгола можно использовать и при изложении алгоритмов для электронных вычислительных машин.

Сокращенный алгол отличается от языка АЛГОЛ-60, в частности, тем, что в нем имеются средства для обозначения частей матриц (строк, граф и т. д.) и для выполнения операций над ними. Например, вычисление суммы парных произведений элементов двух строк или граф записывается с помощью одного оператора.

Потребность в изображении групповых операций отмечается не только нами. В так называемом Новосибирском алголе [20] предлагается ввести целый ряд весьма целесообразных изменений и дополнений в АЛГОЛ-60, в том числе служебные слова **скаляр, вектор, матрица**.

T_A

	1	2	...	n	n+1	n+2	...	n-q	s ₁
1									
1'									
2									
2'									
...									
n									
n'									
n-1									
n-2									
...									
n-q									
s ₁									
s									

T_S

	1	2		n	n-1	n-2	...	n-q	s ₁
1	1								
1'									
2		1							
2'									
...									
n				1					
n'									
n-1					1				
n-2						1			
...									
n-q								1	
s ₁									1

T_3

	1	2	...	n
1				
2				
...				
q				

Рис. 20.

В сокращенном алголе применяются также переменные части идентификаторов, выделяемые круглыми скобками (см. правую часть табл. 1).

Например, изображение массива строки в виде стр $(i)T(k)[1:n]$ указывает, что параметры i и k переменные. Характер изменения параметров может описываться различным способом, в том числе с помощью заголовков циклов.

Введение переменных параметров в идентификатор позволяет в ряде случаев описать одним оператором однотипные операции над различными массивами.

T_A

	1	2		n	S_i
1					
1'					
2					
2'					
n					
n'					
S, S_i					

T_B

	1	2	...	n	S_i
1					
1'					
2					
2'					
...					
n					
n'					
S_i					

T_B
(обратная матрица)

	1	2	...	n
1				
2				
...				
n				

Рис. 21.

В компактных схемах строки (графы) рассматриваются как отдельные массивы. Принимая в рассмотренном примере $i=1$, $k=1$, получим стр 1 $T1[1:n]$ —первую строку табл. 1, принимая $i=2$, $k=3$, получим стр 2 $T3[1:n]$ —вторую строку табл. 3 и т. п.

Процедуры решения систем линейных уравнений, помещенные в табл. 1, основаны на компактной схеме способа Гаусса.

Сущность способа Гаусса состоит в последовательном исключении неизвестных. При этом изменяются начальные значения коэффициентов — к ним последовательно прибавляются произведения чисел, затем производится деление и т. д. Компактная схема отличается от обычной тем, что процесс накопления пар-

T_A									
	1	2	...	n	$n+1$	$n+2$	...	$n+q$	S
1									
1'									
2									
2'									
...									
n									
n'									
S									

T_B					
	1	2	...	n	
1					
2					
...					
q					

T_C					
	1	2	...	n	S
1	1				
1'					
2		1			
2'					
...					
n				1	
n'					
S					1

Рис. 22.

ных произведений объединяется в одну операцию без промежуточных записей. Это очень упрощает расчет на электрических автоматах*.

В процедурах, рассмотренных в табл. 1, проверки можно осуществить и более кратко с помощью служебного слова **проверка**, заменяющего конструкцию **если . . . то . . . иначе . . .**, в тех случаях, когда при несоблюдении контрольного условия повторяется *последний* этап вычислений.

* При решении систем линейных уравнений на ЭВМ применяется другой подход (см. гл. III).

Стандартные процедуры для операторов клавишных автоматов

№ п.п	Инженерные программы	Программы на сокращенном алголе	Содержание процедуры, примечания
1	для $i=1$ до n цикл Начало $M1: k=i$ до S цикл $\sum гр i \times гр k _1^m A \rightarrow$ $\rightarrow i.kB \rightarrow k.iB$ если $\sum стр i _1^{n+q} B \neq$ $\neq i.SB$ то на $M1$ конец i	для $i=1$ до n цикл Начало $M1: k:=i$ до S цикл $B[k i] := B[i, k] :=$ $\sum (гр(i) A[1:m] \times$ $\times гр(k) A[1:m]);$ если $\sum (стр(i) B[1:n +$ $+q]) \neq B[i, S]$ то на $M1$ конец i	Процедура Первая трансформация Гаусса Примечание Бланки табл. А и В (см. рис. 17)
2	для $i=1$ до n цикл Начало $M1: для k=i до S_1 цикл$ $\sum гр i A \times гр k B _1^{S_1} \rightarrow$ $\rightarrow i'.kB;$ если $\sum стр i' B _1^{n+q} \neq$ $\neq i'.S_1 B$ то на $M1$ $M2: -(стр i' _{S_1}^{S_1/i'.i} B \rightarrow$ $\rightarrow стр i' _{S_1} A$ если $\sum стр i' _1^{n+q} A \neq$ $\neq i'.S_1 A$ то на $M2$ конец i	для $i:=1$ до n цикл Начало $M1: для k:=i до S1 цикл$ $B[i', k] :=$ $\sum (гр(i) A[1:S1] \times$ $\times гр(k) B[1:S1]);$ если $\sum (стр(i') B[1:n +$ $+q]) \neq B[i', S1]$ то на $M1;$ $M2: стр(i') A[1:S1] :=$ $-стр(i') B[1:S1] /$ $B[i', i];$ если $\sum (стр i' A[1:n +$ $+q]) \neq A[i, S1]$ то на $M2$ конец i	Процедура Прямой ход решения симметричных систем линейных уравнений Примечание n—порядок, q—число граф свободных членов. Бланки А, Б, В (см. рис. 20). В строке (графе) S_1 дана сумма коэффициентов и свободных членов исходной системы уравнений, в строке S—сумма коэффициентов без свободных членов
3	для $i=1$ до q цикл Начало $M1: для k=n до 1 цикл$ $\sum стр i B \times стр k' A _1^n +$ $+ k'.(n+i) A \rightarrow i.kB;$ если $\sum стр i B \times$ $\times стр S A _1^n +$ $+ S.(n+i) A \neq 0$ то на $M1$ конец i	для $i:=1$ до q цикл Начало $M1: для k:=n до 1 цикл$ $B[i, k] := A[k', n+i] +$ $+ \sum (стр(i) B[1:n] \times$ $\times стр(k' A[1:n]);$ если $\sum (стр(i) B[1:n] \times$ $\times стр(S) A[1:n]) \div$ $+ A[S, n+i] \neq 0$ то на $M1;$ конец i	Процедура Обратный ход для q граф свободных членов

№ п.п	Инженерные программы	Программы на сокращенном алголе	Содержание процедуры, примечания
4	для $i=n$ до 1 цикл Начало $M1$: для $k=i$ до 1 цикл { если $k=i$ то $-m/B[i', i] \rightarrow b$ иначе $0=b$; $b + \sum \text{стр } iB \times$ $\text{стр } k'A \Big _1^n \rightarrow i.kB =$ $= k.iB \}$ если $\sum \text{стр } iB \times$ $\times \text{стр } SA \Big _1^n + m \neq 0$ то на $M1$ конец i	для $i: = n$ до 1 цикл Начало $M1$: для $k=i$ до 1 цикл { если $k=i$ то $b := -m/B[i', i]$ иначе $b := 0$; $B[k, i] := B[i, k] :=$ $:= b + \sum (\text{стр}(i) B[1:n] \times$ $\times \text{стр}(k') A[1:n]) \}$ если $\sum (\text{стр}(i) B[1:n] \times$ $\times \text{стр}(S) A[1:n] + m \neq 0$ то на $M1$; конец i	Процедура Обратный ход для вычисления обратной матрицы (чисел влияния) Примечание Бланки табл. А, Б, В (рис. 21). Прямой ход см. в процедуре 2 при условии, что $q=0$ и $S_1=S$. Коэффициенты обратной матрицы получаются в табл. В умноженными на масштабный коэффициент m, который выбирается из соображений удобства (чтобы числа влияния не были слишком малыми)
5	для $i=1$ до n цикл Начало $M1$: для $k=i$ до S цикл $\sum \text{гр } iA \times \text{гр } kB \Big _1^S \rightarrow$ $\rightarrow i'.kB$; если $\sum \text{стр } i'B \Big _1^n \neq$ $\neq i'.SB$ то на $M1$; $M2$: для $k=i$ до S цикл $-\sum \text{гр } iB \times$ $\times \text{гр } kA \Big _1^S / i'.iB \rightarrow i'.kA$; если $\sum \text{стр } i'A \Big _1^{n+q} \neq$ $\neq i'.SA$ то на $M2$ конец i	для $i: = 1$ до n цикл Начало $M1$: для $k:=i$ до S цикл $B[i', k] :=$ $= \sum (\text{гр}(i) A[1:S] \times$ $\times \text{гр}(k) B[1:S])$; если $\sum (\text{стр}(i') B[1:n]) \neq$ $\neq B[i', S]$ то на $M1$; $M2$: для $k:=i$ до S цикл $A[i', k] :=$ $= -\sum (\text{гр}(i) B[1:S] \times$ $\times \text{гр}(k) A[1:S]) / B[i', i]$; если $\sum (\text{стр}(i') A[1:n +$ $+ q]) \neq A[i', S]$ то на $M2$; конец i	Процедура Прямой ход решения несимметричных систем линейных уравнений Примечание Бланки А, Б, В (рис. 22). Обратный ход см. процедуру 3. Для получения по настоящей программе обратной матрицы следует принять n граф для свободных членов с единицами в клетках 1. $n+1$, 2. $n+2$, ... n. $n+n$.

№ п.п	Инженерные программы	Программы на сокращенном алголе	Содержание процедуры, примечания
6	для $i = 1$ до q цикл Начало $M1$: для $k = 1$ до S цикл $\sum \text{гр} i B_1 \times \text{гр} k B_2 _1^n \rightarrow$ $\rightarrow i.k B_2;$ если $\sum \text{стр} i B_2 _1^n \neq$ $\neq i.S B_2$ то на $M1$ конец i	для $i := 1$ до q цикл Начало $M1$: для $k := 1$ до S цикл $B2[i, k] :=$ $= \sum (\text{гр}(i) B1[1:n] \times$ $\times \text{гр}(k) B[1:n]);$ если $\sum (\text{стр}(i) B2[1:n] \neq$ $\neq B2[i, S])$ то на $M1$; конец i	Процедура Определение величин неизвестных для q граф свободных членов по обратной матрице Примечание Бланки табл. B, B_1, B_2 (рис. 23).

Однако такая замена не всегда возможна. В виде примера рассмотрим предложенный в работе [10] изящный способ расчета некоторых систем с односторонними связями, воспринимающими только сжимающие усилия. После выполнения прямого хода решения системы уравнений сравнивают знаки k и $k+1$ -го преобразованных свободных членов.

T_B						T_{B_1}					T_{B_2}					
	1	2	...	n	s		1	2	...	q		1	2	...	n	s
1						1					1					
2						2					2					
...						...										
n						n					q					

Рис. 23.

Перемена знака указывает, что $k+1$ -ая и последующая связи из работы исключаются, усилия в них принимаются равными нулю и обратный ход выполняется с k -го уравнения.

Процедура 2 для прямого хода остается без изменения (см. табл. 1). Процедура 3 для обратного хода дополняется указанными логическими операциями.

Обратный ход:

для $i=1$ до q цикл

Начало

$1 \rightarrow k$;

M1: если знак $(k+1)' \cdot (n+i)A \neq$ знаку $k' \cdot (n+i)A$, то на M2
иначе $k+1 \rightarrow k$; если $k=n$

то на M2 иначе на M1;

M2: $|\text{стр } 1 + \text{стр } 2 + \dots + \text{стр } k|_1^k A \rightarrow |\text{стр } S^{(i)}|_1^k A$;

$|1 \cdot (n+i) + 2 \cdot (n+i) + \dots + k \cdot (n+i)| A \rightarrow S^{(i)}(n+i)A$;

M3: для $m=k$ до 1 цикл

$\Sigma |\text{стр } iB \times \text{стр } m'A|_1^k + m' \cdot (n+i)A \rightarrow i \cdot mB$;

если $\Sigma |\text{стр } iB \times \text{стр } S^{(i)}A|_1^k + S^{(i)}(n+i)A \neq 0$

то на M3;

конец i .

Отладив приведенную программу на простейшем примере, нетрудно убедиться, что она реализует алгоритм выключения растянутых связей.

По формуле M1 производится сравнение знаков.

Формулы M2 служат для подсчета необходимых при проверках контрольных сумм, располагаемых в строках $S^{(i)}$ (разных для различных комбинаций загрузки).

По формуле M3 реализуется обратный ход, после чего следует проверка.

Если исключить из программы логическую операцию M1, то оператору потребуются дополнительные пояснения, что внесет элемент неопределенности. Практика показала, что при механическом выполнении расчета в этом случае нередко появляются ошибки.

При составлении программ для операторов электрических вычислительных машин имеются две противоположные тенденции. С одной стороны, наибольшей простотой отличается развернутая программа, состоящая из относительно большого числа простейших приказов и проверок, что обеспечивает автоматизм, «бездумность» вычислений [61]. С другой стороны, желательно выразить программу по возможности более кратко. При этом приходится применять некоторые обобщенные понятия (многократное циклирование, старшинство циклов, логические действия и т. д.). При выполнении такой программы оператор должен в уме осуществлять логику счета, подставляя вместо индексов численные значения в соответствии с закономерностями.

ми, выраженными указанными понятиями. Это требует от оператора не только усвоения сравнительно сложных понятий, но и навыков безошибочной их реализации. Вопрос о применении той или иной формы программы должен решаться в зависимости от математической подготовки и склонностей операторов, выполняющих расчет. Нужно учитывать потребность человека в творческой работе и не доводить автоматизм вычислений до крайних пределов. Следует отметить, что чрезмерный автоматизм иногда приводит к неожиданным ошибкам (неправильное чтение частей программы) не контролируемым системой самопроверок.

Не менее острой является проблема ошибок при программировании и подготовке данных для электронных вычислительных машин. В настоящее время постановка задачи на ЭВМ представляет собой многоступенчатый процесс, наиболее уязвимым местом которого являются стыки между ступенями. Именно в стыках возникает наибольшая вероятность ошибок, для избежания которых необходим жесткий контроль: при программировании — предварительная тщательная отладка программы на бумаге и проверка ее вторым лицом, благодаря чему отладка на машине осуществляется с одной-двух пробных попыток; при решении численной задачи на ЭВМ — обязательное независимое дублирование и сверка подготовки исходных данных, записи их на бланки и перфорирования.

Однако при большой исходной информации даже добросовестная проверка не всегда достигает цели — ошибки все же иногда бывают и обнаруживаются лишь при анализе окончательных результатов. Кардинальное решение проблемы состоит в полной автоматизации всех ступеней процесса и ликвидации стыков между ними: автоматизации программирования, разработке устройств непосредственного чтения, отображения цифрового и графического материала и др.

Глава III

О РЕШЕНИИ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ СТРОИТЕЛЬНОЙ МЕХАНИКИ НА ЭВМ

ИСПОЛЬЗОВАНИЕ ОСОБЕННОСТЕЙ СТРУКТУР МАТРИЦ ДЛЯ ПОВЫШЕНИЯ ВОЗМОЖНОГО ПОРЯДКА СИСТЕМ, РЕШАЕМЫХ НА ЭВМ

При расчете сложных статически неопределимых конструкций (стержневых систем, тонкостенных конструкций типа пластинок и оболочек, вантовых систем, пространственных массивов в гидротехнических сооружениях и т. п.) инженеру приходится сталкиваться со значительными вычислительными трудностями.

Одной из главных является проблема решения систем линейных уравнений высоких порядков.

Имеющиеся стандартные программы в зависимости от размеров оперативной памяти машины — 1024, 2048 и 4096 ячеек — позволяют соответственно решать несимметричные системы уравнений 30, 42, 64 порядков и симметричные системы уравнений 40, 59, 86 порядков.

Как видим, здесь имеются определенные ограничения. Между тем, нередко сложные задачи расчетов сооружений, особенно пространственные задачи, приводятся к системам линейных уравнений со многими десятками и даже сотнями неизвестных. Поэтому очень важно разработать приемы, позволяющие повысить возможный порядок системы, вмещающейся в оперативную память машины.

Ниже предлагается использовать определенные свойства матриц, наиболее часто встречающихся в задачах строительной механики и прикладной теории упругости. Обычно система уравнений располагается диагональной полосой, лентой (рис. 24). Это так называемые ленточные матрицы. Все элементы, расположенные вне ленты, являются нулевыми. Если ограничивающие ленту линии не параллельны главной диагонали (пунктирные линии на рис. 24), то лента дополняется нулевыми элементами до линий, параллельных главной диагонали. Строка матрицы может быть значительных размеров (40—80 клеток), но в системах высоких порядков незаполненные углы матрицы остаются все же достаточно большими.

Характерной особенностью ленточной матрицы является то, что после прямого хода решения системы уравнений методом исключения (Гаусса) структура преобразованной матрицы по-прежнему остается ленточной, нулевые элементы, расположенные вне ленты, в процессе решения не участвуют и остаются нулевыми.

Характер структуры матрицы может быть отражен в программе, что достигается введением специального параметра — порядка ленты.

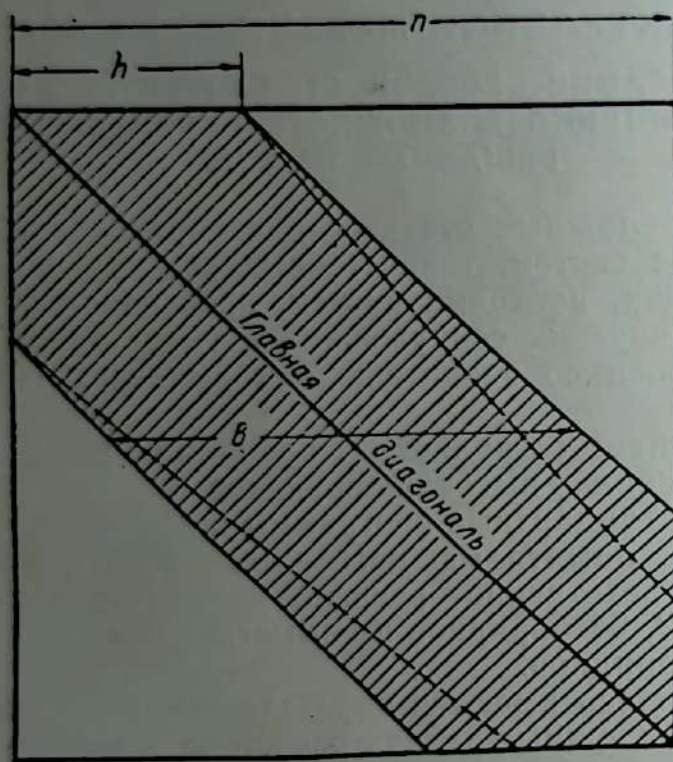


Рис. 24.

Порядком ленты h будем называть число клеток строки, расположенных справа от главной диагонали, включая главный элемент.

Размер строки b связан с порядком ленты такой зависимостью:

$$b = h + h_2,$$

где h_2 — число клеток слева от главной диагонали.

Наименьший порядок ленты $h=1$ имеет место в диагональной матрице, наибольший порядок ленты $h=n$ — в полностью заполненных матрицах.

Таким образом любая система уравнений, в том числе и полностью заполненная, может трактоваться как ленточная.

Между предельными значениями порядков ($h=1$ и $h=n$) располагаются все типы ленточных матриц:

$h=2, b=3$ — трехчленные матрицы;
 $h=3, b=5$ — пятичленные матрицы;
 $h=4, b=7$ — семичленные матрицы;
 $h=5, b=9$ — девятичленные матрицы и т. п.

В оперативную память машины записываются только элементы, расположенные внутри ленты; нулевые элементы, расположенные вне ленты, исключаются.

Нумерация ячеек для несимметричной матрицы показана на рис. 25 на примере системы порядка $n=9$ и порядка ленты $h=4$.

		$n=9$										
		$h=4$										
1											Свободные элементы	
0		1	2	3	4							52
1		5	6	7	8	9						53
2		10	11	12	13	14	15					54
3		16	17	18	19	20	21	22				55
4			23	24	25	26	27	28	29			56
5				30	31	32	33	34	35	36		57
6					37	38	39	40	41	42		58
7						43	44	45	46	47		59
8							48	49	50	51		60

Рис. 25.

Нумерация ячеек для симметричной матрицы показана на рис. 26; элементы, расположенные ниже главной диагонали, в этом случае в память не записываются.

Программа для ленточных матриц отличается от обычной программы составлением некоторых дополнительных логических условий. Так, для симметричной матрицы количество элементов в строке не должно превышать величины $h_1=h$, а со строки $n-h$ (первая строка имеет нулевой номер) величина h_1 при переходе к каждой последующей строке уменьшается на единицу.

В данном примере (см. рис. 26) для строк:

$$5 \dots h_1 = h = 4$$

$$6 \dots h_1 = 4 - 1 = 3$$

$$7 \dots h_1 = 3 - 1 = 2$$

$$8 \dots h_1 = 2 - 1 = 1$$

Имеются также определенные зависимости между адресами диагональных элементов — каждый последующий элемент отличается от предыдущего на величину h_1 . Аналогичные зависимости имеют место и для несимметричных матриц. Указанные зависимости были использованы для составления команд формирования адресов и логических проверок окончания этапов.

прования адресов и номеров

		$n=9$								Свободные члены
		$h=4$								
1	2									
0	1	2	3	4						31
1		5	6	7	8					32
2			9	10	11	12				33
3				13	14	15	16			34
4					17	18	19	20		35
5						21	22	23	24	36
6							25	26	27	37
7								28	29	38
8									30	39

Рис. 26.

Аналогично построены все известные программы решения систем линейных уравнений, но отличие состоит во введении в программу нового параметра — порядка ленты h , что позволяет резко повысить возможный порядок n решаемой системы.

Объем ячеек оперативной памяти, необходимый для решения симметричной системы уравнений при одной графе свободных членов, составляет

$$v = n(h+1) - 0,5h(h-1); \quad (1)$$

для несимметричной системы уравнений —

$$v = 2nh - h(h-1). \quad (2)$$

Отсюда можно получить возможный порядок решаемой системы при полезном объеме оперативной памяти v (за вычетом ячеек, занимаемых программой).

Для симметричной системы

$$n = \frac{v + 0,5h(h-1)}{h+1}. \quad (3)$$

Для несимметричной системы

$$n = \frac{v + h(h-1)}{2h}. \quad (4)$$

Например, при объеме памяти 1024 ячейки и полезном объеме $v=900$ (121 ячейку занимает программа) можно на ЭВМ «Киев» решить симметричную систему трехчленных уравнений ($h=2$) 300 порядка, пятичленных ($h=3$)—225 порядка, девятичленных* ($h=5$)—91 порядка и т. д., тогда как по обычной программе порядок решаемой симметричной системы не превышает 40.

Еще более значителен эффект при использовании машин с большим объемом оперативной памяти—в 2048—4096 ячеек. В этом случае порядок системы повышается даже при весьма широких лентах, характерных для задач, решаемых методом сеток.

Так, при объеме памяти 4096 ($v=3900$) и порядке ленты $h=25$ (размер строки $25+24=49$ клеток) можно решить симметричную систему уравнений 161 порядка:

$$n = \frac{3900 + 0,5 \times 25(25-1)}{25+1} = 161.$$

Даже в тех случаях, когда система «идет» по обычной программе, предлагаемая программа (назовем ее ленточной) обладает определенными преимуществами — упрощается подготовка исходных данных и объем вводимой информации, поскольку

* К неполным девятичленным уравнениям перестановкой неизвестных приводятся восьмичленные уравнения цилиндрических оболочек В. З. Власова [58].

ку автоматически исключаются нулевые элементы, расположенные вне ленты.

Программа построена так, что графы свободных членов вводятся последовательно и автоматически до тех пор, пока их число не станет равно заданной величине q .

Счет по программе идет настолько быстро, что печать результатов решения для отдельных граф свободных членов происходит с секундными перерывами, то есть практически непрерывно. Особенностью является также то, что программа в зависимости от признака α может выдавать как обратную матрицу (при $\alpha=0$), так и значения неизвестных для q граф свободных членов (при $\alpha=q$).

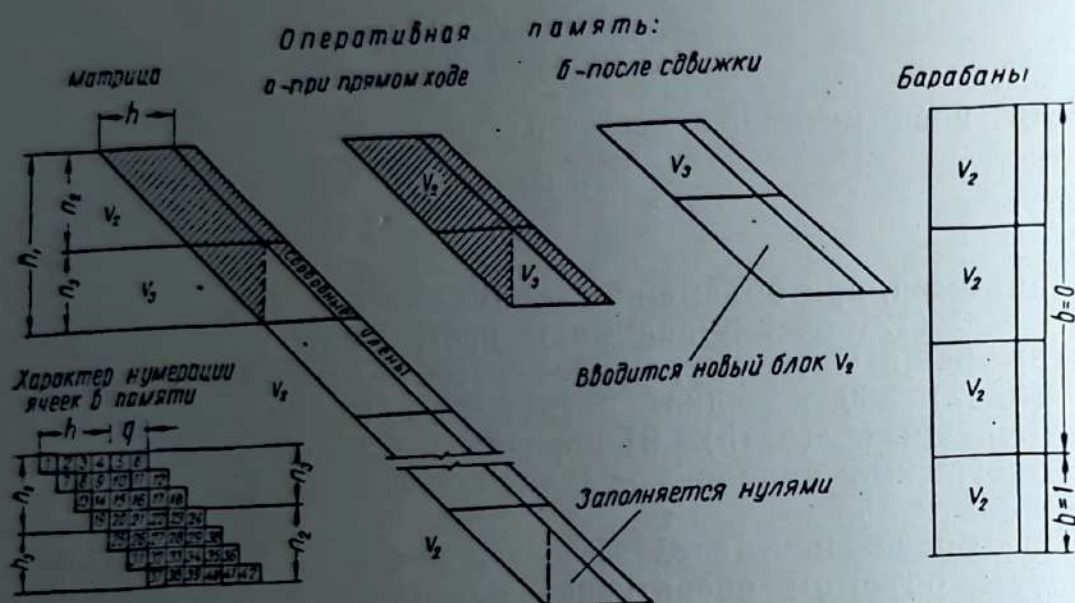


Рис. 27.

Ленточная запись матрицы позволяет весьма эффективно использовать барабаны (внешнюю память машины), особенно для симметричных систем. После исключения при прямом ходе группы неизвестных n_2 (рис. 27) преобразовываются массивы чисел v_2 и частично v_3 , остальная часть матрицы ниже заштрихованной части остается без изменения. Это позволяет вводить матрицу в оперативную память машины блоками. Сначала вводится блок $v_1 = v_2 + v_3$. После выполнения прямого хода и исключения n_2 неизвестных блок v_2 направляется в барабаны. Затем блок v_3 сдвигается (рис. 27, б), на освободившееся место засылается новая порция перфокарт v_2 и снова выполняется прямой ход. В результате последовательного выполнения прямого хода в оперативной памяти и засылки в барабаны блоков v_2 вся преобразованная матрица оказывается в барабанах. Далее матрица

направляется частями из барабанов в оперативную память, выполняется обратный ход и вычисляются неизвестные. После печати каждой порции неизвестных производится их сдвигка в памяти подобно тому, как это делалось при прямом ходе, но не вверх, а вниз.

Программа предусматривает возможность либо введения в машину блоков матрицы, либо формирования этих блоков в оперативной памяти по специальной программе в тех случаях, когда матрица состоит из однотипных уравнений. При этом используется операторный способ записи (см. стр 82).

Величина n_3 (см. рис. 27) равна $h-1$; n_1 — максимальный порядок матрицы, который можно поместить в оперативную память. Отсюда $n_2 = n_1 - n_3 = n_1 - (h-1)$. Следовательно, если порядок ленты h больше n_1 , то воспользоваться предлагаемым способом уже нельзя. По изложенной схеме в оперативную память вводится не только заштрихованная часть матрицы, но и дополнительная часть так, чтобы блок u_3 имел форму параллелограмма. Это упрощает прямой ход, но увеличивает необходимое количество ячеек по сравнению с получаемым по формуле (1) и уменьшает возможный порядок n_1 (второго члена в числителе формулы (3) в этом случае нет). Поэтому был разработан второй вариант программы, схема которого сводится к следующему: в память вводится блок u_2 , нижняя часть матрицы в памяти (блок u_3), имеющая форму треугольника, очищается. Выполняется прямой ход до $i = n_2 - 1$. В результате в ячейках треугольника u_3 записываются числа, представляющие собой поправки к исходным коэффициентам. Далее блок u_2 направляется в барабаны, а на освободившееся место в верхней части матрицы засылается новый блок u_2 . После этого производится сдвигка треугольного блока u_3 вверх с одновременным суммированием с соответствующими числами блока u_2 , а в освободившиеся ячейки блока u_3 засылаются нули. Далее снова выполняется прямой ход и цикл повторяется.

Второй вариант программы сложнее первого, но допускает введение в машину более широких лент.

Особенность предлагаемых программ решения систем линейных уравнений с использованием барабанов состоит в том, что в отличие от известных программ, предназначенных для той же цели, они основаны не на групповом, а на обычном исключении неизвестных. Это обеспечивает большую простоту программы и меньший объем вычислений *. Но, как указывалось, имеются известные ограничения в отношении порядка ленты, кото-

* В работе [16] показано, что групповое исключение неизвестных целесообразно для трехклеточных матриц (клетка — группа коэффициентов), приводится весьма эффективная программа для решения с использованием барабанов систем такого вида, а также указывается возможность создания аналогичной программы для пятиклеточных систем.

рые тем меньше, чем больший объем оперативной памяти машины.

Изложенный прием составления программ решения систем линейных уравнений можно применить, не только к ленточным, но и к любым другим матрицам, преобразованная структура которых известна.

Так составлена программа для ленточной окаймленной матрицы (рис. 28), которая при преобразовании сохраняет свою

	$n = 9$										
	$h = 3$			$n_1 = 4$							
i	0	1	2	3					4	5	Свободные члены
	0	1	2	3					4	5	36
	1		6	7	8				9	10	37
	2			11	12	13			14	15	38
	3				16	17	18		19	20	39
	4					21	22	23	24	25	40
	5						26	27	28	29	41
	6							30	31	32	42
	7								33	34	43
	8									35	44

Рис. 28.

структуру. Здесь вводится, помимо порядка n системы, еще два параметра: порядок ленты h и порядок пустого треугольника n_1 . Логика программы, естественно, несколько сложнее, а программа длиннее, чем для ленточной матрицы, изображенной на рис. 26.

Характерной особенностью рассмотренных программ является то, что позволяя значительно повысить порядок во многих частных случаях, они являются вполне общими.

Так, по программе, составленной для ленточной окаймленной матрицы, можно решить полностью заполненную матрицу, если принять $n=h$ и $n_1=0$.

Рассмотрим второй путь, который может быть применен при всех циклических методах расчета (к ним относятся методы

последовательных приближений, скорейшего спуска, сопряженных градиентов). Для всех этих методов [63] характерно то, что в процессе решения матрица коэффициентов остается неизменной, а все вычисления выполняются в рабочих ячейках, количество которых линейно зависит от порядка матрицы.

Здесь можно воспользоваться тем, что в матрицах высоких порядков многие строки подобны, то есть при сдвиге одной строки относительно другой на некоторое количество клеток получается полное совпадение, а также тем, что в каждой строке много нулевых элементов.

В оперативную память записываются только операторы, характеризующие группы подобных строк.

Для каждой строки и имеющей тот же номер графы матрицы отводится специальная адресная ячейка-указатель (рис. 29). В первом и третьем адресе этой ячейки указаны адреса операторов, к которым относится данная строка и графа матрицы.

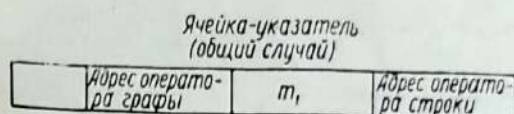


Рис. 29.

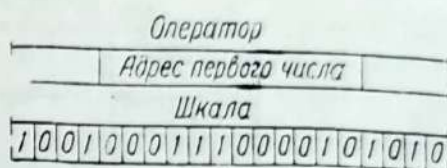


Рис. 30.

В частном случае симметричной матрицы для графы и строки указан адрес одного и того же оператора. Во втором адресе зафиксирован номер m_1 начального элемента строки (графы). В начальной ячейке оператора (рис. 30) указан адрес первого

*Ячейка-указатель
(для строки ленточной симметричной матрицы)*



Рис. 31.

значащего числа строки (графы), в последующих одной или нескольких ячейках-шкалах содержится информация о расположении значащих чисел в строке (графе). Единица соответствует значащему числу, нуль — нулевому элементу.

Значащие числа располагаются отдельным массивом*. Пользуясь шкалой, можно сформировать строку (графу) мат-

* Вначале располагали значащие числа вместе с оператором, к которому они относятся [62]. Но это оказалось неудобным, так как в одном массиве оказались числа в разных системах исчисления — восьмеричной (шкалы) и десятичной (числа), а это не давало возможности применить подпрограмму перевода десятичных чисел в двоичные ко всему массиву в целом.

рицы, последовательно выделяя содержимое каждого разряда с помощью операции логического умножения (см. стр. 36) и, если оно равно единице, заслать в соответствующее место памяти значащее число, в противном случае — нуль.

Для симметричной матрицы, имеющей ленточную структуру, адрес оператора можно поместить во втором адресе ячейки указателя (рис. 31). Вся остальная информация не нужна.

ОПТИМАЛЬНЫЕ ПРОГРАММЫ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ ИСКЛЮЧЕНИЯ

При решении систем линейных уравнений на ЭВМ отказываемся от компактных схем и применяем обычное последовательное исключение неизвестных*. Соответствующая эскизная программа имеет следующий вид (рис. 32).

Матрица

	1	2	3	...	n-2	n-1	n	q
1								
2								
3								
...								
n-2								
n-1								
n								
a								

Рис. 32.

Прямой ход

Для несимметричной системы

$$|\text{стр } k|_{i+1}^q - \frac{k.i}{l.i} \times |\text{стр } i|_{i+1}^q \rightarrow |\text{стр } k|_{i+1}^q$$

для $i=1$ до n , $k=i+1$ до q .

* В главе I было показано, что компактная схема, целесообразная при решении систем уравнений на электрических автоматах, не имеет преимуществ при решении на ЭВМ.

Для симметричной системы уравнений можно адрес k, i заменить i, k и не записывать коэффициенты, расположенные ниже главной диагонали:

$$|\text{стр } k|_{i+1}^q - \frac{i \cdot k}{i \cdot i} \times |\text{стр } i|_{i+1}^q \rightarrow |\text{стр } k|_{i+1}^q.$$

После выполнения прямого хода получаем на месте нахождения исходной преобразованную треугольную матрицу (для несимметричной системы две треугольные матрицы — верхнюю и нижнюю).

Обратный ход

$$-\frac{\sum |\text{стр } a \times \text{стр } (n-i)| + (n-i) \cdot q}{(n-i) \cdot (n-i)} \rightarrow a \cdot (n-i)$$

для $i=0$ до $n-1$.

Пользуясь изложенными эскизными программами и учитывая ленточность, получим ряд программ для ЭВМ, которые в записи на алголе приведены ниже. Программы 1, 2, 3 даны в краткой форме с ориентацией на программирующую программу*, программа 4 дана в развернутой форме.

Программа 1 решения несимметричных систем линейных уравнений с ленточной структурой

Начало Примечание n, h — порядки системы, ленты, i, D, m — номера, соответственно, строк, диагональных элементов, элементов в строке, $v = (2h-1) \times n - h \times (h-1)$ — количество элементов в ленточной матрице, $A[1 : v+n]$ — матрица и вектор свободных членов, рассматриваемые как линейный массив (ввиду сложной структуры матрицы). После первого этапа прямого хода (признак $\beta=0$) на месте нахождения исходной получаем преобразованную матрицу. Нижняя часть ее, расположенная под главной диагональю, служит для преобразования свободных членов (второй этап прямого хода по признаку $\beta \neq 0$), верхняя часть служит для определения неизвестных (обратный ход). $q=0$ — признак обращения матрицы, $q \neq 0$ — признак вычисления неизвестных для q граф свободных членов. Ввод в оперативную память каждого последующего столбца свободных членов выполняется программно на место, занимаемое предыдущим столбцом, после печати полученных в нем величин неизвестных. Чтение $(A, b1, b2)$, печать $(A, b1, b2)$ — нестандартные процедуры ввода данных и вывода результатов от $A[b1]$ до $A[b2]$;

* Развернутую адресную запись этих программ см. в работе [62].

Целые $n, h, q, Q, v, h_1, h_2, D_0, D_1, r, i, m, m_1, j, j_1, j_2, \beta$;
 действительный массив $A[1 : v+n]$;
 чтение $(A, 1, v)$; $\beta := 0$;
 $M1: h_1 := h; h_2 := D_0 := 1$;
 для $i := 0$ шаг 1 до $n-2$ цикл

Начало прямой ход:

$M2: j_1 := j_2 := 0; D_1 := D_0$;
 для $m_1 := 1$ шаг 1 до h_1-1 цикл

Начало

$M3: D_1 := D_1 + h_1 + h_2 + j_1 - j_2$;
 $r := D_1 - m_1$; если $\beta = 0$ то на $M4$;
 $A[Q+i+m_1] := A[Q+i+m_1] - A[r] \times A[Q+i]/A[D_0]$;
 на $M5$;

$M4$: для $m := 1$ шаг 1 до h_1-1 цикл
 $A[r+m] := A[r+m] - A[r] \times A[D_0+m]/A[D_0]$;

$M5$: если $h > i + m_1 + 1$ то $j_1 := j_1 + 1$;
 если $i + m_1 > n - h$ то $j_2 := j_2 + 1$

конец m_1

$D_0 := D_0 + h_1 + h_2$;
 если $h-1 > i+1$ то $h_2 := h_2 + 1$;
 если $i+1 > n-h_1$ то $h_1 := h_1 - 1$

конец i

если $\beta \neq 0$ то на $M10$; $Q := D_0 + 1$; $j := \beta := 1$;
 если $q = 0$ то на $M8$;
 $M6$: чтение $(A, Q, Q+n-1)$; на $M1$;
 $M7: j := j+1$; если $j \leq q$ то на $M6$; на стоп;
 $M8$: для $m := 1$ шаг 1 до n цикл $A[D_0+m] := 0$;
 $A[D_0+j] := 1$; на $M1$;
 $M9: j := j+1$; если $j \leq n$ то на $M8$; на стоп;
 $M10: r := D_0 + n$; $i := 1$;
 $M11: A[r] := -A[r]/A[D_0]$; если $D_0 = 1$ то на $M12$; если $h_1 \neq h$
 то $h_1 := h_1 + 1$;

$D_0 := D_0 - h_1 - h_2$; $i := i+1$;
 если $i > n - h + 1$ то $h_2 := h_2 - 1$; $r := r - 1$;
 для $m := 1$ шаг 1 до $h_1 - 1$ цикл $A[r] := A[r] + A[D_0 + m] \times$
 $\times A[r+m]$; на $M11$;

$M12$: печать $(A, Q, Q+n-1)$;
 на если $q = 0$ то $M9$ иначе $M7$;

стоп:

конец

Программа 2 решения симметричных систем линейных уравнений с ленточной структурой

Начало Примечание n, h —порядки системы, ленты, $v = n \times h - 0,5 h \times (h-1)$ —количество ячеек, расположенных выше главной диагонали матрицы. Остальные примечания те же, что в программе 1;

Целые $n, h, h1, q, v, D0, D1, r, j, j1, m1, m, Q, \beta$;
действительный массив $A[1 : v+n]$;

чтение $(A, 1, v)$; $\beta := 0$;

M1: $h1 := h$; $D0 := 1$;

для $i := 0$ шаг 1 до $n-2$ цикл

Начало прямой ход:

M2: $j := 0$; $D1 := D0$;

для $m1 := 1$ шаг 1 до $h1-1$ цикл

Начало

M3: если $\beta = 0$ то на M4;

$A[Q+i+m1] := A[Q+i+m1] - A[D0+m1] \times A[Q+i]/A[D0]$;

на M5;

M4: $D1 := D1 + h1 - j$;

для $m := 0$ шаг 1 до $h1-m1-1$ цикл

$A[D1+m] := A[D1+m] - A[D0+m1] \times A[D0+m1+m]/A[D0]$;

M5: если $n-h1 \leq m1+i-1$ то $j := j+1$

конец $m1$

$D0 := D0 + h1$;

если $n-h1 < i$ то $h1 := h1-1$

конец i

если $\beta \neq 0$ то на M10; $Q := D0+1$; $j1 := \beta := 1$;

если $q=0$ то на M8;

M6: чтение $(A, Q, Q+n-1)$; на M1;

M7: $j1 := j1+1$; если $j1 \leq q$ то на M6; на стоп;

M8: для $m := 1$ шаг 1 до n цикл $A[D0+m] := 0$;

$A[D0+j1] := 1$; на M1;

M9: $j1 := j1+1$; если $j1 \leq n$ то на M8; на стоп;

M10: $r := D0+n$;

M11: $A[r] := -A[r]/A[D0]$; если $D0=1$ то на M12;

если $h > h1$ то $h1 := h1+1$; $D0 := D0-h1$;

$r := r-1$;

для $m := 1$ шаг 1 до $h1-1$ цикл

$A[r] := A[r] + A[D0+m] \times A[r+m]$;

на M11;

M12: печать $(A, Q, Q+n-1)$;

на если $q=0$ то на M9 иначе M7

конец

**Программа 3 решения симметричных систем
линейных уравнений с ленточной окаймленной структурой**

Начало Примечание $n, h, n1$ —порядки системы, ленты, пустого треугольника, $v=0,5 (n^2-n1^2+n-n1)$ —количество ячеек, расположенных выше главной диагонали матрицы;

Целые $n, h, n1, q, h1, v, r, r1, i, j, j1, m1, m, Q, \beta$;
Действительный массив $A[1:v+n]$;

чтение $[A, 1, v]; \beta := 0$;

$M1: h1 := n - n1; D0 := 1$;

для $i := 0$ шаг 1 до $n-2$ цикл

Начало прямой ход:

$M2: j := 0; D1 := 0; v := 0$;

для $m1 := 1$ шаг 1 до $h1-1$ цикл

Начало

$M3: \text{если } \beta = 0 \text{ то на } M4; \text{ если } m1 \geq h \text{ то если } n1 \geq i \text{ то}$
 $v := n1 - i$;

$r1 := Q + m1 + v + i$;

$A[r1] := A[r1] - A[D0 + m1] \times A[Q + i] / A[D0]$;

на $M7$;

$M4: D1 := D1 + h1$; если $i + m1 + v \leq n1$ то

на $M5$; $j := j + 1; D1 := D1 - j + 1$;

$M5: \text{если } m1 \leq h-1 \text{ то на } M6; v := v + 1$;

если $v + i \leq n1$ то на $M4$;

$M6: \text{для } m := 0 \text{ шаг 1 до } h1 - m1 - 1 \text{ цикл}$

Начало

$r := D1 + m$; если $m1 + m > h-1$ то $r := r + m1 - j$;

$A[r] := A[r] - A[D0 + m1] \times A[D0 + m1 + m] / A[D0]$;

конец m

$M7:$

конец $m1$

$D0 := D0 + h1$; если $i \geq n1$ то $h1 := h1 - 1$;

конец i ,

если $\beta \neq 0$ то на $M12$; $Q := D0 + 1$;

$j1 := \beta := 1$; если $q = 0$ то на $M10$;

$M8: \text{чтение } (A, Q, Q + n - 1)$; на $M1$;

$M9: j1 := j1 + 1$; если $j1 \leq q$ то на $M8$;

на стоп;

$M10: \text{для } m := 1 \text{ шаг 1 до } n \text{ цикл } A[D0 + m] := 0$;

$A[D0 + j1] := 1$; на $M1$;

$M11: j1 := j1 + 1$; если $j1 \leq n$ то на $M10$;

на стоп;

M12: $r := D0 + n$; $v := 0$;

M13: $A[r] := -A[r]/A[D0]$;

если $D0 = 1$ то на M14; если $n - n1 \neq h1$ то $h1 := h1 + 1$ иначе $v := v + 1$;

$D0 := D0 - h1$; $r := r - 1$;

для $m := 1$ шаг 1 до $h1 - 1$ цикл

Начало

$r1 := r + m$; если $h1 \geq n - n1$ то

если $m > h - 1$ то $r1 := r1 + v$;

$A[r] := A[r] + A[D0 + m] \times A[r1]$;

конец m ; на M13;

M14: печать (A , Q , $Q + n - 1$);

на если $q = 0$ то M11 иначе M9;

стоп: конец

Программа 4 решения симметричных систем линейных уравнений ленточной структуры — первый вариант использования барабанов

Начало Примечание Принципиальные предпосылки программы изложены в первом разделе данной главы (см. рис. 27).

h —порядок ленты, q —число граф свободных членов,

$up(v0)$ —количество ячеек, занимаемых программой прямого (обратного) хода,

$v1$ —количество ячеек матрицы, которое можно разместить в оперативной памяти (полезный объем памяти),

$v2$ —объем блока, пересылаемого в барабаны,

$v3$ —блок, сдвигаемый в оперативной памяти.

$v4, v5, v6$ —количество ячеек, занимаемых соответственно операторами, указателями, значащими числами,

$n1, n2, n3$ —количество строк соответственно в блоках $v1, v2, v3$.

Величина $n1$ определяется по формуле

$$n1 = n2 + n3 = \frac{v1}{h + q}.$$

Ограничение: программу можно применять при $n1 > h$.

Идентификаторы массивов: $A0$ —программы и операторов, $A1$ —указателей, $A2$ —значащих чисел, A —матрицы. $\omega0, \omega1, \omega2$ —количество блоков $v2$, размещаемых соответственно в барабанах $B0, B1, B2$.

B, ω — переменные величины,

b — порядковые номера барабанов,

c — номер последнего используемого барабана.

Программу можно использовать и без привлечения внешней памяти (барабанов). В этом случае принимается $c = \omega = 0$.

γ — признак, от которого зависит ввод либо всех чисел матрицы (при $\gamma = 0$), либо операторов и указателей с последующим формированием матрицы (при $\gamma = 1$). Значение γ задается перед решением задачи.

$r1$ — рабочая ячейка, в первом разряде которой записана единица, а в остальных — нули,

$r2$ — рабочая ячейка, в одном из разрядов которой расположена единица, а в остальных — нули (единица перемещается).

Ячейки $r1$ и $r2$ служат для выделения содержимых разрядов шкалы, то есть информации о размещении значащих чисел в операторе.

Формирование матрицы, прямой и обратный ход, сдвигка при обратном ходе выделены в отдельные процедуры. Это облегчает обозрение и чтение основной программы и заменяет блок-схему.

Нестандартные процедуры:

чтение $\Pi(A, a1, a2)$ — ввод с перфокарт,

запись $(A, a1, a2)$ на $(B, b1, b2)$ — пересылка на барабаны,

чтение $(A, a1, a2)$ с $(B, b1, b2)$ — ввод с барабанов,

печать $(A, a1, a2)$ — печать результатов.

Формальные параметры следующие: A, B — идентификаторы массивов в памяти и на барабанах, $a1, a2$ и $b1, b2$ — границы массивов соответственно в памяти и на барабанах.

vN — количество ячеек, занимаемых нестандартными процедурами чтения, печати, перевода из десятичной системы исчисления в двоичную и обратно, обращения к интерпретирующей системе машины и т. п.

Решение задачи начинается вводом с пульта программ нестандартных процедур, прямого хода и относящихся к нему процедур, операторов и значащих чисел, то есть вводом массивов $A0[1 : vN + v\Pi + v4]$ и $A2[1 : v6]$.

Если формирование матрицы не предусмотрено ($\gamma = 0$), то принимается $v4 = 0$ и массив значащих чисел $A2[1 : v6]$ не вводится. В этом случае ввод блоков матрицы выполняется программно; Целые $vN, v\Pi, v0, v1, v2, v3, v4, v5, v6, n1, n2, n3, h, q, b, a, k, j, \gamma, i, D0, D1, v, g, m1, m, r, r1, r2, r3, v, v0, v1, v2, B, B0, B1, B2$;

Действительные S, a ;

Целые массивы $A0[1 : v\Pi + v4], A1[1 : v5]$;

Действительные массивы $A2[1 : v6], A[1 : v1]$;

Процедура формирование:

Начало

- 1: если $a=0$, то на 3;
- 2: $i := 1$; $\mu := 1$; на 5;
- 3: $i := n3 + 1$;
- 4: $\mu := n3 \times (h+q) + 1$;
- 5: $v := 1$;
- 6: $g := 0$;
- 7: $r2 := r1$;
- 8: $r3 := A0[A1[i] + v] \wedge r2$;
- 9: если $r2 = r3$ то на 11;
- 10: $A[\mu] := 0$; на 13;
- 11: $A[\mu] := A2[A0[A1[i]] + g]$;
- 12: $g := g + 1$;
- 13: $\mu := \mu + 1$;
- 14: $r2 := r2/2$;
- 15: если $\mu = i \times (h+q) + 1$ то на 18;
- 16: если $r2 \neq 0$ то на 8;
- 17: $v := v + 1$; на 7;
- 18: $i := i + 1$;
- 19: если $i \leq n1$ то на 5;

Конец формирования

Процедура прямой ход (n)

Начало целое n ;

- 1: $D0 := 1$;
- 2: $i := 0$;
- 3: $D1 := D0$;
- 4: $m1 := 1$;
- 5: $m := 0$;
- 6: $v := 0$;
- 7: $D1 := D1 + h + q$;
- 8: $A[D1 + m] := A[D1 + m] - A[D0 + m1] \times A[D0 + m1 + m] / A[D0]$;
- 9: $m := m + 1$;
- 10: если $m1 + m \neq h$ то на 8;
- 11: $A[D1 + h + v] := A[D1 + h + v] - A[D0 + m1] \times A[D0 + m1 + m] / A[D0]$;
- 12: $m := m + 1$;
- 13: $v := v + 1$;
- 14: если $m1 + m \neq h + q$ то на 11;
- 15: $m1 := m1 + 1$;
- 16: если $m1 \neq h$ то на 5;
- 17: $D0 := D0 + h + q$;
- 18: $i := i + 1$;
- 19: если $i \neq n$ то на 3;

Конец прямого хода

Основная программа:

- 1: $v1 := (h+q) \times n1$; $v2 := (h+q) \times n2$;
 $n3 := n1 - n2$; $v3 := v1 - v2$;
- 2: $b := 0$; $a := 1$;
- 3: если $b=0$ то на 6;
- 4: если $b=1$ то на 7;
- 5: если $b=2$ то на 8;
- 6: $B := B0$; $\omega := \omega0$; на 9;
- 7: $B := B1$; $\omega := \omega1$; на 9;
- 8: $B := B2$; $\omega := \omega2$;
- 9: $j := 0$;
- 10: если $\gamma=0$ то на 13;
- 11: чтение $\Pi(A1, 1, v5)$;
- 12: формирование; на 16;
- 13: если $a=0$ то на 15;
- 14: чтение $\Pi(A, 1, v1)$; на 16;
- 15: чтение $\Pi(A, v3+1, v1)$;
- 16: $a := 0$;
- 17: если $b=c$ то на 24;
- 18: если $j=\omega$ то на 23;
- 19: прямой ход ($n2$);
- 20: запись $(A, 1, v2)$ на $(B, j \times v2 + 1, j \times v2 + v2)$;
- 21: $A[1:v3] := A[v2+1:v1]$;
- 22: $j := j+1$; на 10;
- 23: $b := b+1$; на 4;
- 24: если $j \neq \omega$ то на 19;
- 25: прямой ход ($n1$);

Примечание На этом заканчивается программа циклов формирования (или ввода) матрицы и прямого хода. Ниже следует программа обратного хода, вводимая на место программы прямого хода;

- 26: чтение $\Pi(A0, vH+1, vH+v0)$;
Процедура обратный ход ($i1, n$):

Начало целые $i1, n$;

- 1: $i := i1$;
- 2: $D1 := D0 - i1 \times (h+q)$;
- 3: $g := 0$;
- 4: $S := 0$;
- 5: $r := D1$;
- 6: $m := 1$;
- 7: $r := r + h + q$;
- 8: если $r > D0$ то на 12;
- 9: $S := S + A[D1+m] \times A[r+h+g]$;
- 10: $m := m+1$;
- 11: если $m \neq h$ то на 7;

```

12:  $A[D1+h+g] := -(A[D1+h+g]+S)/A[D1]$ ;
13:  $g := g+1$ ;
14: если  $g \neq q$  то на 4;
15:  $i := i+1$ ;
16: если  $i = n1$  то на 18;
17:  $D1 := D1-h-q$ ; на 3;
18:  $i := 0$ ;
19:  $g := 1$ ;
20:  $a := A[h+g+i \times (h+q)]$ ;
21: печать ( $A, a$ );
22:  $g := g+1$ ;
23: если  $g \leq q$  то на 20;
24:  $i := i+1$ ;
25: если  $i \neq n$  то на 19;
конец обратного хода

```

Примечание Если машина имеет буферное устройство, то величины a засылаются на буфер, после чего производится печать массива с буфера.

Процедура сдвигка

Начало

```

1:  $i := 0$ ;
2:  $m := 0$ ;
3:  $r := i \times (h+q) + m$ ;
4:  $A[v1-r] := A[v3-r]$ ;
5:  $m := m+1$ ;
6: если  $m \neq q$  то на 3;
7:  $i := i+1$ ;
8: если  $i \neq h-1$  то на 2;
конец сдвигки

```

Основная программа:

```

1: обратный ход (0,  $n1$ );
2: если  $j=0$  то на 7;
3:  $j := j-1$ ;
4: сдвигка;
5: чтение ( $A, 1, v2$ )  $c(B, j \times v2+1, j \times v2+v2)$ ;
6: обратный ход ( $n1-n2, n2$ ); на 2;
7: если  $b=0$  то на стоп;
8:  $b := b-1$ ;
9: если  $b=1$  то на 11;
10:  $B := B0$ ;  $\omega := \omega0$ ; на 2;
11:  $B := B1$ ;  $\omega := \omega1$ ; на 2;
стоп:

```

конец программы

Глава IV

ПОЛОГИЕ ПРОСТРАНСТВЕННЫЕ КОНСТРУКЦИИ

ПРИМЕРЫ ИЗ ПРАКТИКИ ПРОЕКТИРОВАНИЯ, АНАЛИЗ РАСЧЕТНЫХ СХЕМ

Ниже рассмотрены задачи расчетно-теоретического и практического характера применительно к конструкциям, запроектированным автором в НИИ экспериментального проектирования б. АСнА УССР для ряда сооружений.

Конструкция монолитной железобетонной пологой оболочки двоякой кривизны (рис. 33). Оболочка прямоугольной формы в плане размерами 18×26 м, стрелой подъема 4,4 м и толщиной плиты 6 см. По контуру она опирается на железобетонные бортовые элементы, расположенные в вертикальных плоскостях. Шаг колонн вдоль бортовых элементов 2 м. Оболочка построена в Киеве над рестораном «Метрополитен» (станция «Крещатик» Киевского метрополитена) в 1961 г.

В центральной зоне (область безмоментной работы) оболочка заармирована конструктивной арматурой в виде сетки с квадратными ячейками. Угловые зоны оболочки армируются более мощной арматурой, воспринимающей главные растягивающие усилия и идущей по направлению, перпендикулярному к диагоналям прямоугольного плана оболочки. В зонах, примыкающих к бортовым элементам, арматура располагается двойной сеткой для восприятия тех изгибающих моментов, которые возникают в этих зонах. В этих местах, а также в угловых зонах оболочка имеет большую толщину, чем в центральной зоне.

Расчетная схема оболочки имеет следующие особенности: оболочка относится к классу пологих оболочек двоякой кривизны и обладает анизотропностью вследствие переменной толщины бетона и неодинакового армирования ее отдельных зон; существенным фактором является неодинаковая работа железобетона на сжимающие и растягивающие усилия. Кроме того, оболочка имеет сложные граничные условия—железобетонные бортовые

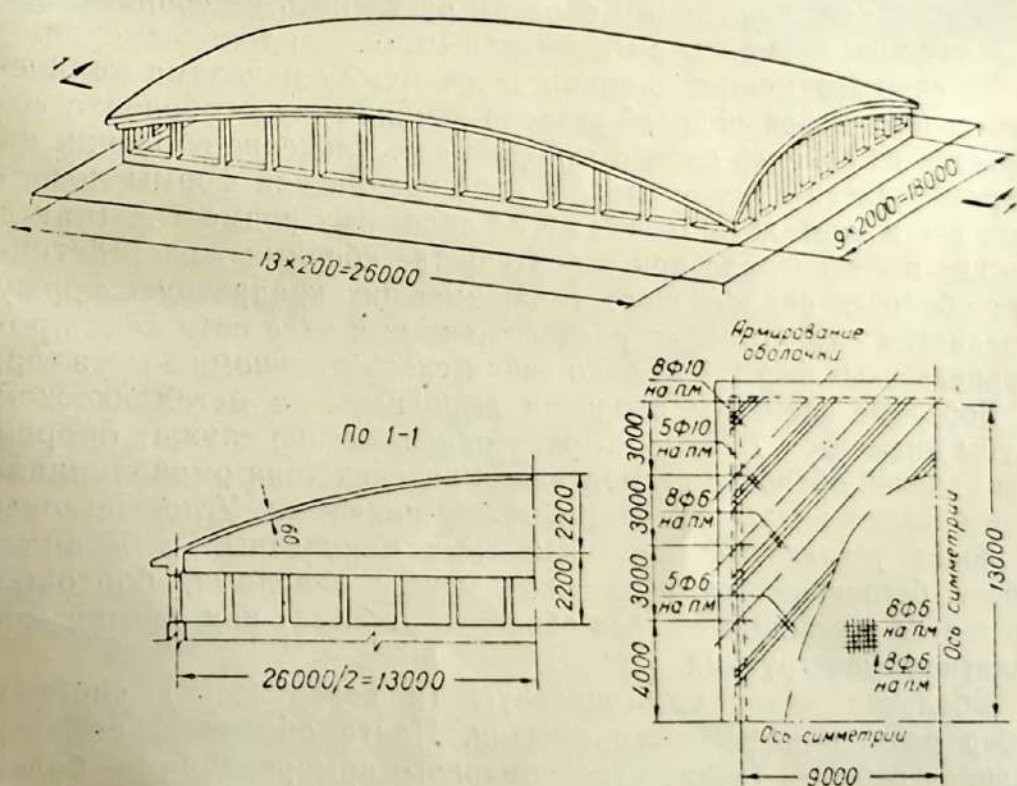
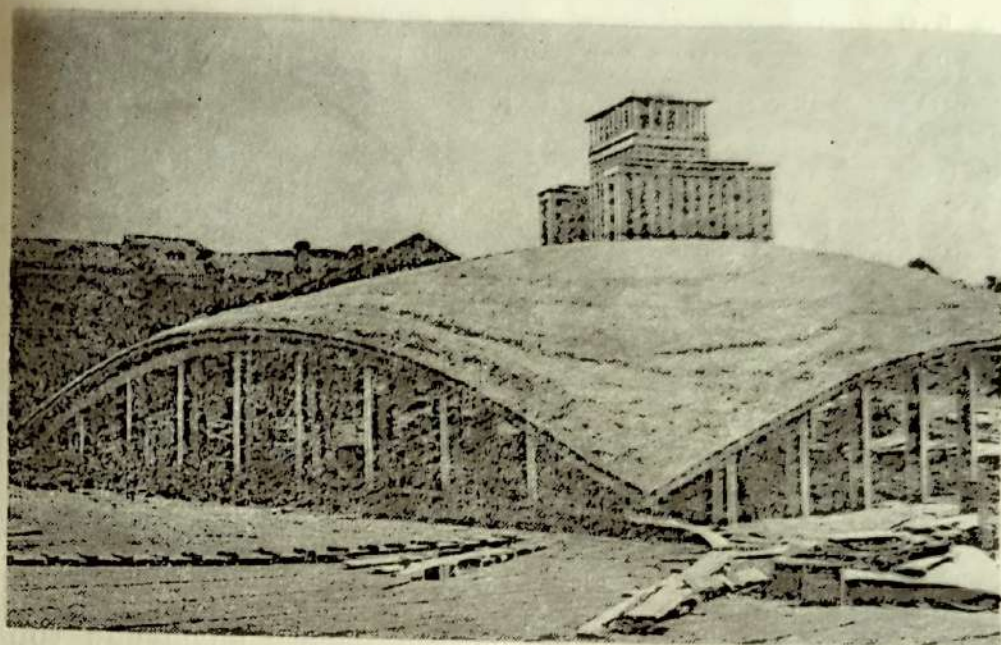


Рис. 33.

элементы, на которые она опирается по контуру, работают на осевое растяжение и изгиб в вертикальной плоскости. Ввиду того, что бортовые элементы упруго податливы, каждая точка контура оболочки имеет три линейных смещения, которые определяются с учетом совместной работы оболочки с бортовыми элементами.

Сборная железобетонная пологая оболочка двоякой кривизны (рис. 34). Оболочка предназначена для покрытия центральной части Киевского аэровокзала (г. Борисполь), строительство ее находится в стадии завершения. Оболочка имеет близкую к прямоугольной форму в плане с размерами 48×58 м и стрелой подъема в центре 8,9 м. Она выполнена из сборных железобетонных ребристых панелей размерами $2,6 \times 8,0$ м, с высотой продольных и поперечных ребер соответственно 40 и 30 см и толщиной плиты 4 см (в угловых зонах толщина плиты панелей 6 см).

Срединная поверхность оболочки образуется частью поверхности тора, две стороны которого ограничены вертикальными плоскостями, две другие — наклонными. Угол наклона таков, что наклонные плоскости пересекаются на оси вращения тора. Часть поверхности выбрана из соображений, чтобы поле оболочки можно было наилучшим образом расчленить на сборные железобетонные элементы с минимальным числом типоразмеров.

За счет переменной ширины швов между панелями количество типоразмеров панелей сведено к минимуму — оболочка собирается из панелей одного типоразмера. Членение оболочки на сборные панели прямоугольной, а не квадратной формы позволило вести монтаж оболочки не на сплошных подмостях (как в Москве и Ленинграде при строительстве сборных железобетонных оболочек аналогичного типа, имевших квадратную форму в плане), а на подмостях, расположенных в виде пяти лент пространственных ферм. Это дало значительную экономию металла.

Бортовые элементы оболочки выполнены из металлической трубы диаметром 720 мм, которая одновременно служит опорой для панелей оболочки на монтаже и впоследствии омоноличивается бетоном вместе со швами между панелями. Угловые зоны оболочки усиливаются путем местных монолитных набетонок. После бетонирования всех швов между панелями, бортовых элементов и угловых зон вся оболочка работает как единая монолитная конструкция.

Оболочка имеет ортогональную (в поверхности) систему ребер разного поперечного сечения. Плита оболочки, подкрепленная ребрами (ввиду неравномерного армирования — более мощного в угловых зонах и примыкающих к бортовым элементам), анизотропна. В ограничивающих плоскостях она опирается на упругие бортовые элементы, работающие на осевое растяжение с изгибом в вертикальной плоскости.

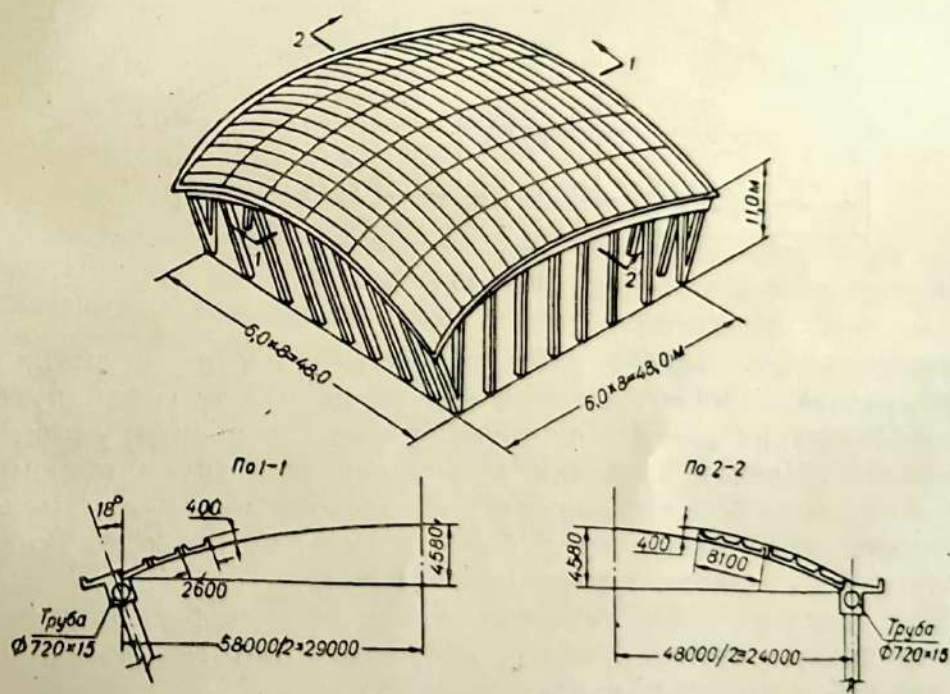
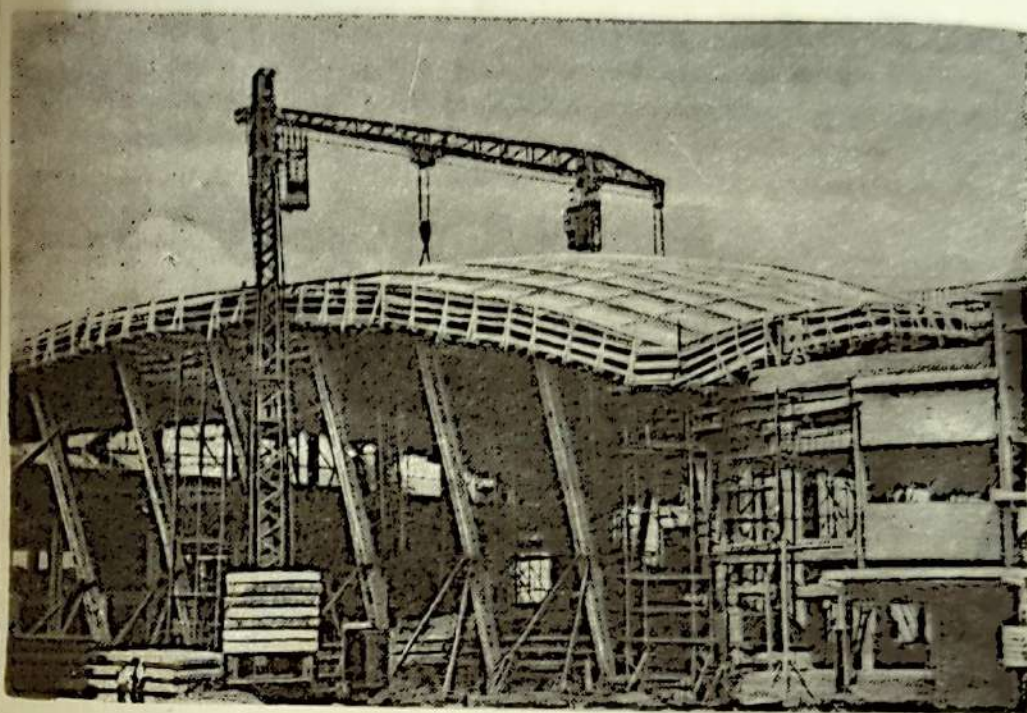


Рис. 34.

Сборная железобетонная цилиндрическая оболочка (рис. 35), опертая по двум прямым сторонам, с металлическими затяжками, расположенными вдоль этих сторон с шагом 3,0 м. Она предназначена для покрытия однопролетных и многопролетных одноэтажных промышленных и сельскохозяйственных зданий. Оболочка образована из сборных

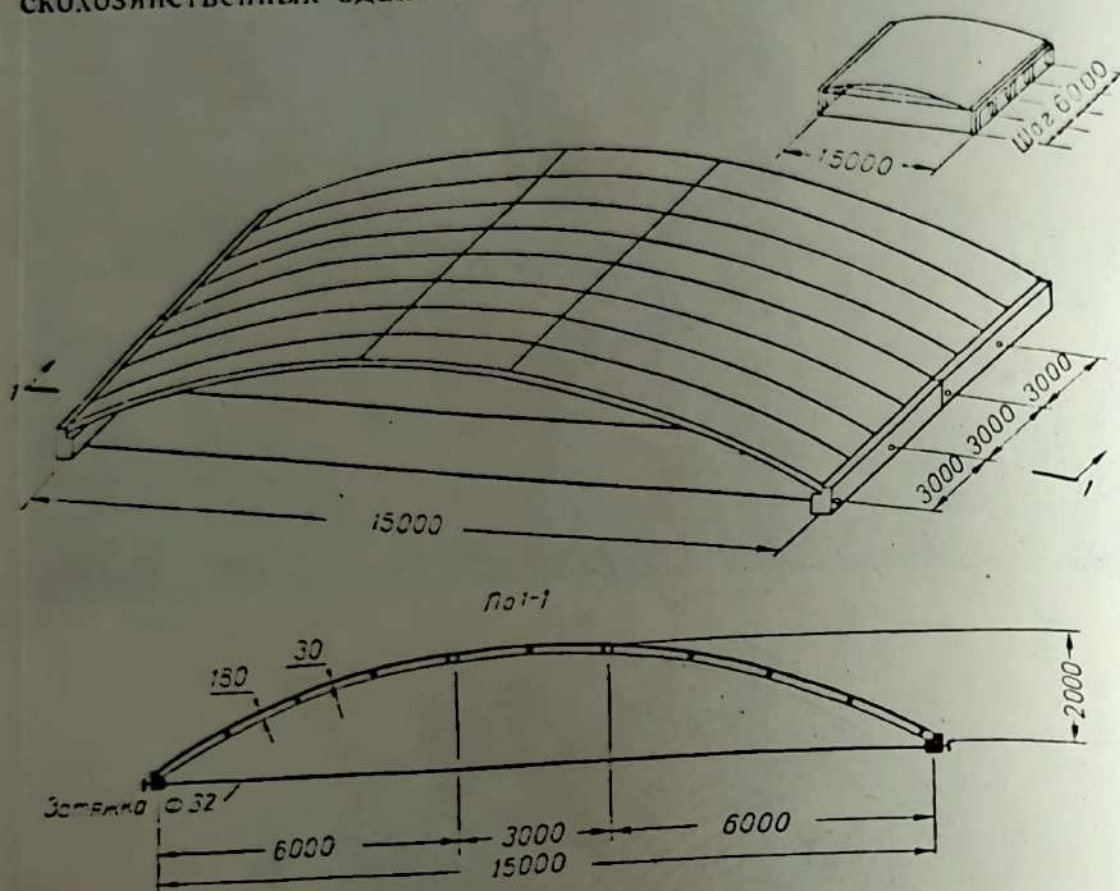


Рис. 35.

железобетонных панелей двух типоразмеров (два крайних и один средний элемент), которые стыкуются между собой путем замоноличивания швов. Конструкция швов, перпендикулярных к прямым сторонам оболочки, шарнирная (конструкция безмоментного стыка более экономична по расходу металла и менее трудоемка, чем моментного). Оболочка имеет ортогональную систему ребер разного поперечного сечения.

Как видно из конструктивной схемы, оболочка относится к классу пологих цилиндрических оболочек с ортогональной системой ребер, имеющих различную жесткость.

Граничные условия оболочки таковы, что она имеет свободное опирание по двум прямым сторонам и закреплена перпендикулярно к этим сторонам упругими затяжками. Два других края оболочки являются свободными.

Конструкция покрытия для одноэтажных промышленных и сельскохозяйственных зданий (рис. 36). Элементы покрытия представляют собой сборные железобетонные фермы, образующие пространственный шатер решетчатого типа, который опирается в четырех углах на сборные железобетонные колонны. Сетка колонн 6×6 м. Фермы устанавливаются на колонны каркаса наклонно (с небольшой вспа-

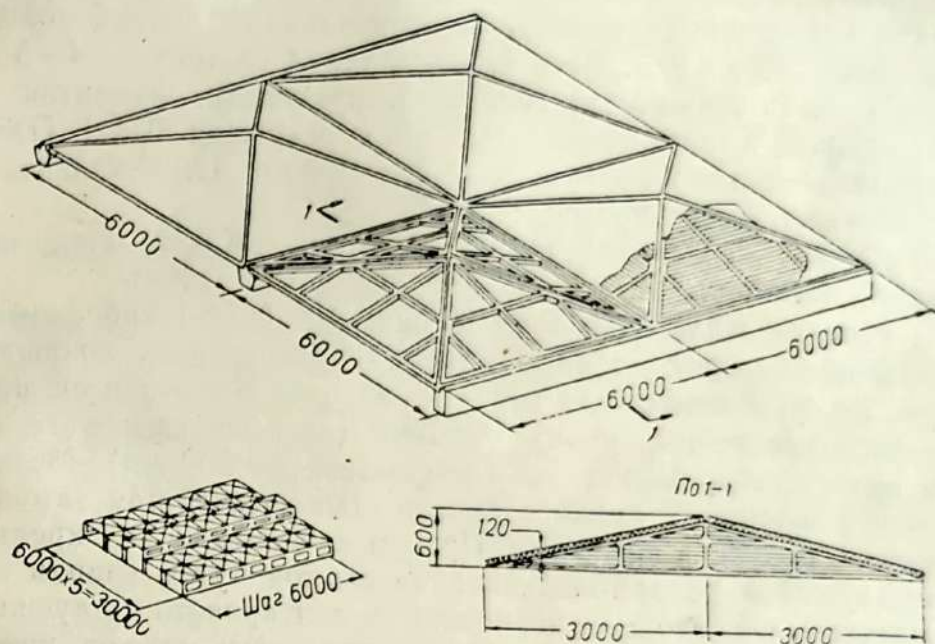


Рис. 36.

рушенностью — не более 1 м) так, что из четырех ферм образуется шатер на одну ячейку в виде правильной четырехгранной пирамиды. Закладные детали стыка свариваются. Поля между решеткой фермы заполняются в заводских условиях (одновременно с изготовлением самой фермы) плитами утеплителя, которые являются самонесущей ограждающей конструкцией. Такая конструкция совмещенного покрытия дает возможность сократить расход стали и бетона на 1 м^2 перекрытия. Весь каркас здания состоит из трех типоразмеров конструктивных элементов, включая фундаменты.

С расчетной точки зрения эта конструкция покрытия представляет собой пологую решетчатую пространственную систему с нерегулярной структурой решетки. Каждая грань пирамиды может рассматриваться как система перекрестных изгибаемых в вертикальной плоскости балок, опертых на треугольный в плане контур, образованный верхним и нижним поясами ферм. На контуре эти балки жестко соединены с диагональными и крайними элементами, которые являются для них упругими опорами.

Кроме того, диагональные ребра шатра, контурные ребра и перпендикулярные к ним элементы решетки работают на осевые усилия.

Конструкция сборного железобетонного безригельного каркаса (рис. 37), предназначенного в качестве основной несущей конструкции для жилых и гражданских зданий каркасного типа. Сборные железобетонные элементы каркаса разработаны для сетки колонн 6×6 м. Аналогичного типа конструкции разработаны для зданий с сеткой колонн $5,4 \times 5,4$ или $4,8 \times 4,8$ м. Весь каркас здания состоит из 4—5 (в зависимости от этажности) типов конструктивных элементов:

1) сборная железобетонная панель перекрытия ПЖ-1, ПЖ-2 (покрытия)—2 типоразмера, которые можно изготавливать в одной опалубочной форме;

2) сборная железобетонная колонна КЖ-1, КЖ-2—1 или при большой этажности 2 типоразмера;

3) сборный железобетонный фундамент Ф-1—1 типоразмер.

Сборные железобетонные панели перекрытий и покрытия ребристые, расположенные ребрами кверху. Высота всех продольных и поперечных ребер — 250 мм (при меньшем шаге колонн высота ребер может быть уменьшена до 200 мм). Между собой и с колоннами каркаса панели стыкуются путем замоноличивания выпусков арматуры. Причем на монтаже они крепятся между собой до замоноличивания стыков путем сварки закладных деталей. Это дает возможность монтировать следующие этажи, не ожидая, пока бетон замоноличенных стыков нижележащих перекрытий наберет требуемую прочность. В плане панели имеют размеры 3×6 м. Они достаточно удобны в транспортировке и монтаже.

После замоноличивания стыков между панелями каждое перекрытие работает в условиях, мало отличающихся от условий работы монолитного безбалочного перекрытия, и обладает всеми достоинствами такого перекрытия при сохранении полной сборности и простоты стыков (стыки работают безмоментно и представляют собой фактически цилиндрические шарниры). Такая работа сборного перекрытия достигается тем, что стыки панелей расположены по линиям, близким к траекториям нулевых моментов. Такое перекрытие имеет лучшие технико-экономические показатели по сравнению с другими аналогичного типа сборными перекрытиями. Устойчивость здания обеспечивается (при небольшой этажности) рамностью всей конструкции каркаса — стыки панелей перекрытий с колоннами жесткие. При большой этажности (более 9 этажей) устойчивость и прочность на ветровую нагрузку обеспечивается железобетонными диафрагмами между колоннами или вертикальными металлическими связями. Роль диафрагм могут выполнять стены лестничных клеток и лифтов при их соответствующем конструктивном решении.

С расчетной точки зрения каркас представляет собой сложную многопролетную и многоярусную пространственную раму с тонкими плитами перекрытий в качестве ригелей, подкрепленных системой взаимно-перпендикулярных ребер. Ввиду неравномерного армирования ребра имеют разную изгибную жесткость. Кроме того, в плите перекрытий по определенным направлениям расположены цилиндрические шарниры.

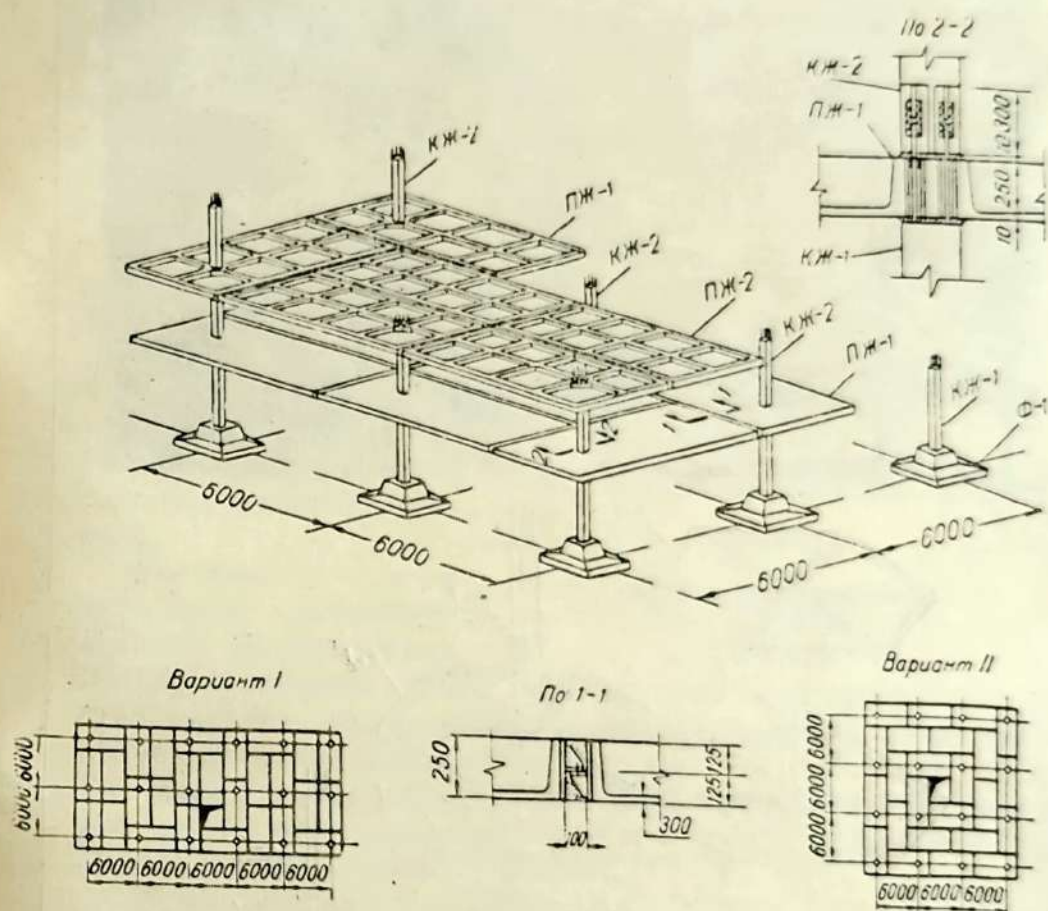


Рис. 37.

Конструкция сборного железобетонного безригельного каркаса (рис. 38), но с иной разрезкой перекрытий на сборные панели. Эта конструкция, известная под названием «грибовидной», применена в здании летней гостиницы в Каневе, построенной в 1961 г. Сборные железобетонные элементы каркаса разработаны для сетки колонн $3,6 \times 3,6$ м. Весь каркас здания состоит из трех типов конструктивных элементов:

- 1) сборная железобетонная панель П-1 перекрытий и покрытия одного типоразмера;
- 2) сборная железобетонная колонна К-1 одного типоразмера;

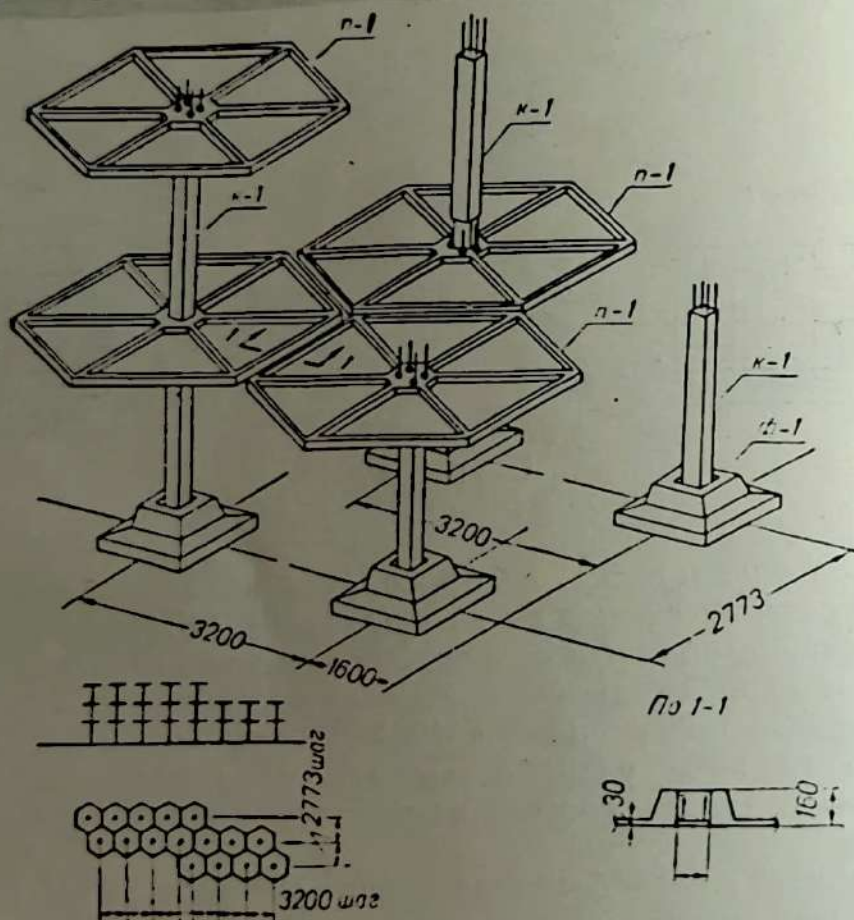


Рис. 38

3) сборный железобетонный фундамент Ф-1 одного типоразмера.

Сборные железобетонные панели перекрытий и покрытия имеют шестиугольную форму в плане и расположены ребрами кверху. Высота контурных и радиальных ребер 180 мм. Стыкуются панели путем замоноличивания выпусков арматуры из плит. Так как шаг колонн сравнительно невелик, для сокращения количества типоразмеров панелей и др. при членении перекрытий линии стыков между панелями целесообразно расположить несколько в стороне от траекторий нулевых моментов. После замоноличивания стыков перекрытие работает (как и в предыдущем случае) как монолитное безбалочное и обладает всеми его достоинствами.

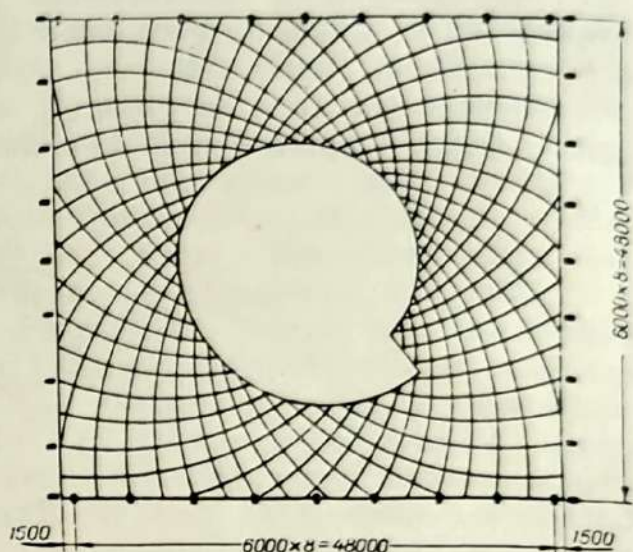


Рис. 39.

Расчетная схема каркаса — многопролетная многоэтажная

пространственная рама с тонкими плитами перекрытий в качестве ригелей, подкрепленными системой ребер с нерегулярной структурой в плане. Особенностью работы описанных конструкций каркасов является также то, что рамы каркасов имеют разные расчетные схемы в период монтажа конструкций в период эксплуатации готового здания.

Монолитное железобетонное перекрытие размерами в плане 48×48 м (рис. 39) *. В центре перекрытие имеет отверстие улиткообразной формы, по контуру которого перекрытие опирается на металлические стойки трубчатого сечения. По наружному контуру оно опирается на сборные железобетонные колонны каркаса здания.

Конструктивно перекрытие решено в виде тонкой железобетонной плиты толщиной 6 см, подкрепленной ребрами высотой 60 см, образующими систему перекрестных криволинейных балок. Чтобы можно было рационально заармировать балки перекрытия, они направлены по траекториям, где моментный инвариант распределяется между двумя направлениями поровну.

* Это перекрытие расположено на отм. 4,5 м в центральной части Киевского аэровокзала (г. Борисполь).

С расчетной точки зрения перекрытие представляет собой тонкую плиту, подкрепленную системой криволинейных ребер. Система этих ребер имеет в плане нерегулярную структуру. Кроме того, плита и подкрепляющие ее ребра обладают анизотропностью ввиду неравномерного армирования.

Ниже приведены примеры висячих железобетонных покрытий для перекрытия больших пролетов общественных зданий.

Висячие покрытия применяются в строительной практике сравнительно недавно, но уже имеются интересные решения таких конструкций. Дальнейшее их изучение, разработка практически приемлемых методов расчета, выявление наиболее экономичных форм покрытий и т. д. являются в настоящее время актуальными проблемами для инженеров и ученых.

Обладая всеми достоинствами обычных железобетонных оболочек, такие покрытия имеют еще ряд существенных преимуществ. Общая устойчивость вследствие особенности геометрии поверхности и характера статической работы не имеет для них такого решающего значения, как для обычных оболочек. Это дает возможность делать покрытия больших пролетов достаточно тонкими, а значит и сравнительно экономичными. Бортовые элементы, в которых сосредоточена значительная часть материала, работают в основном на сжатие. Для создания вантовой сети можно использовать высокопрочную проволоку, не применяя громоздких и сложных устройств для предварительного натяжения арматуры. Сборные железобетонные конструкции несложны в изготовлении и имеют небольшое число типоразмеров. Монтаж конструкций прост и не требует дополнительных подмостей.

Особенностью всех железобетонных висячих покрытий является то, что на стадии монтажа такие конструкции работают как вантовые сети, воспринимающие нагрузку от собственного веса плит, а при эксплуатации — как железобетонные оболочки, воспринимающие всю эксплуатационную нагрузку.

Вантовая сеть должна обеспечивать удобство и легкость монтажа, то есть допускать возможность навески на нее монтажных подмостей и других монтажных приспособлений. Для этого вантовая сеть должна обладать первоначальной жесткостью, которая достигается путем придания вантовой сети такой геометрии, при которой каждая нить оказывается пригруженной другими нитями. Иногда для увеличения жесткости вантовой сети полезно вводить тросы жесткости — ванты, способные воспринимать не только вертикальные, но и горизонтальные составляющие от примыкающих к ним вант. Ванты, которые имели раньше кривизну разных знаков, разделяются тросами жесткости на отдельные участки, натяжения в которых могут быть разными не только по величине, но и по знаку.

Вантовая сеть имеет хорошие статические характеристики в том случае, когда равновесная нагрузка для нее близка к вер-

тикальной равномерно распределенной нагрузке. Равновесной нагрузкой для вантовой сети (нити) называется такая нагрузка, которую может воспринять данная вантовая сеть (нить) без изменения своей геометрии (имеется в виду, что упругие удлинения настолько малы, что ими можно пренебречь).

Так, например, для крученой поверхности (гиперболический параболоид), все нити которой направлены по параболам, одним из видов равновесной нагрузки является равномерно распределенная. Следует отметить, что жесткая и экономичная вантовая сеть, как правило, образуется при натяжении на несимметричный контур, который требует устройства тяжелых бортовых элементов распорных опор. С другой стороны, более экономичный

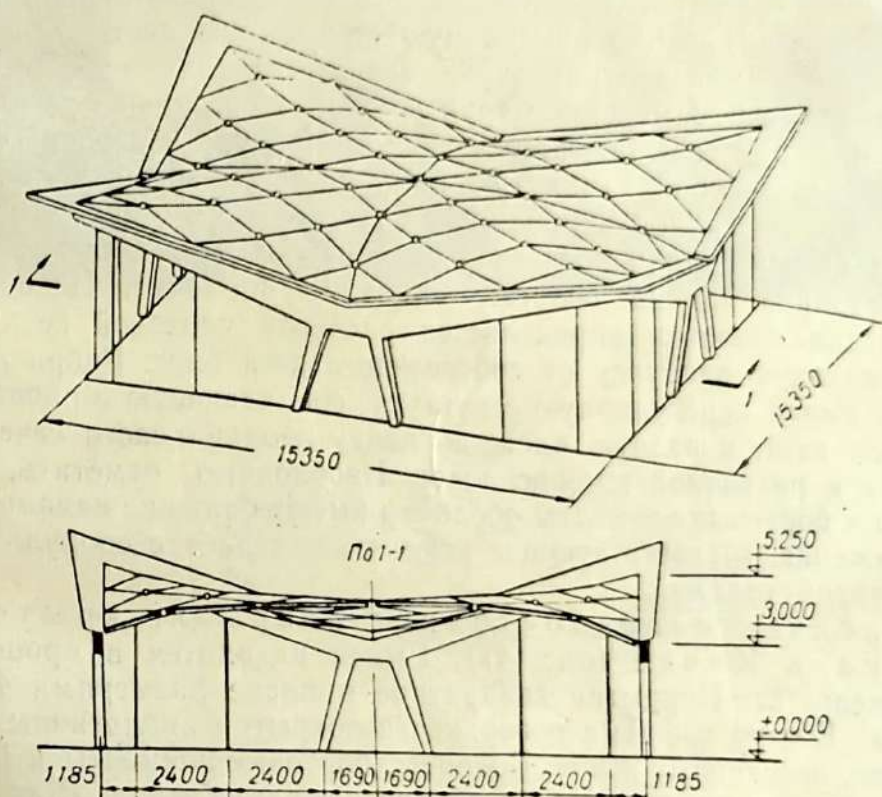


Рис. 40.

и легкий симметричный контур приводит к ухудшению вантовой поверхности, появлению сплошных зон, большим неравномерностям в распределении усилий между вантами и т. д. Эти противоречия обуславливают большое разнообразие форм висячих покрытий.

Висячее железобетонное покрытие торгового павильона (рис. 40). Квадратное в плане покрытие имеет поднятые четыре угловые точки. Борт покрытия собран из сборных железобетонных элементов, которые установлены на

сборные железобетонные колонны и металлические стойки, расположенные по контуру здания. Стыки сборных элементов борта — моментные. Между бортовыми элементами параллельно диагоналям квадрата натягиваются ванты из арматурных стержней. На полученную таким образом самонапряженную вантовую сеть навешиваются сборные железобетонные плиты двух типоразмеров, а после замоноличивания швов между ними создается сборно-монолитное железобетонное висячее покрытие. Приведенная толщина покрытия не превышает 4 см.

На первой стадии расчетная схема покрытия представляет собой самонапряженную вантовую сеть, натянутую на контуре, который образован бортовыми элементами. С точки зрения строительной механики вантовая сеть представляет собой геометрически мгновенно изменяемую стержневую систему, которая, воспринимая вертикальную нагрузку от плит, значительно изменяет свою геометрию. Структура такой системы нерегулярна, а площади сечений вант разные. Контур вантовой сети представляет собой замкнутую раму сложной конфигурации с переменной изгибной и продольной жесткостями стержней.

На второй стадии расчетной схемой покрытия является железобетонная тонкая пологая ребристая оболочка. Исходная геометрия оболочки определяется расчетом вантовой сети на вертикальную нагрузку от собственного веса плит. Ребра оболочки имеют нерегулярную структуру, совпадающую с расположением вант, и разную площадь ввиду неодинакового сечения вантов и различной ширины швов. Необходимо отметить, что ванты и бортовые элементы оболочки имеют большие начальные напряжения, соответствующие усилиям при работе сооружения по первой стадии.

Висячее железобетонное покрытие крытого рынка в Киеве (рис. 41). Рынок находится в процессе строительства. Покрытие квадратное в плане размерами 42×42 м. Контур и форма поверхности покрытия аналогичны покрытию, описанному выше. Колонны, бортовые элементы и плиты покрытия выполняются из сборного железобетона. Шаг колонн, расположенных по периметру здания, 6 м. Колонны жестко защемлены в фундаментах и бортовых элементах покрытия. Приведенная толщина бетона покрытия 5 см.

Расчетная схема такого покрытия на первой и второй стадиях во многом соответствует расчетной схеме покрытия, конструкция которого приведена выше. Отличие заключается в том, что вантовая сеть имеет более густую сетку, а следовательно, и более густую сетку ребер железобетонной оболочки. Кроме того, железобетонные колонны, жестко связанные с сжато-изогнутым бортом, также работают на изгиб.

Висячее железобетонное покрытие крытого рынка в г. Черкассы. Строительство крытого рынка начато

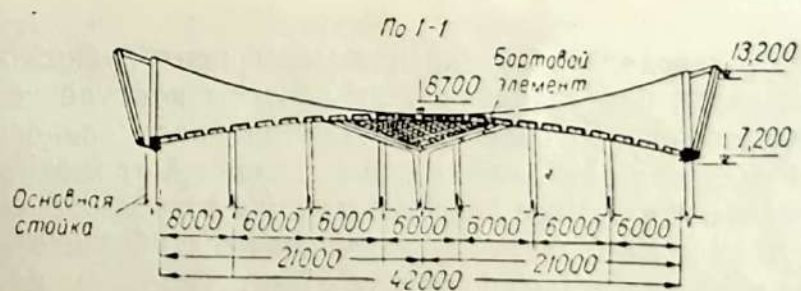
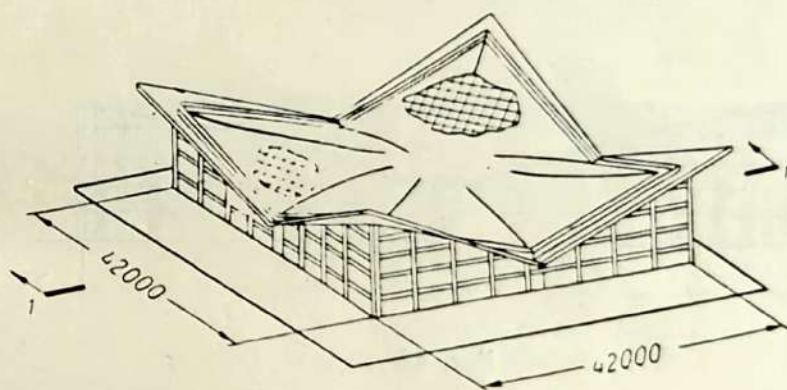
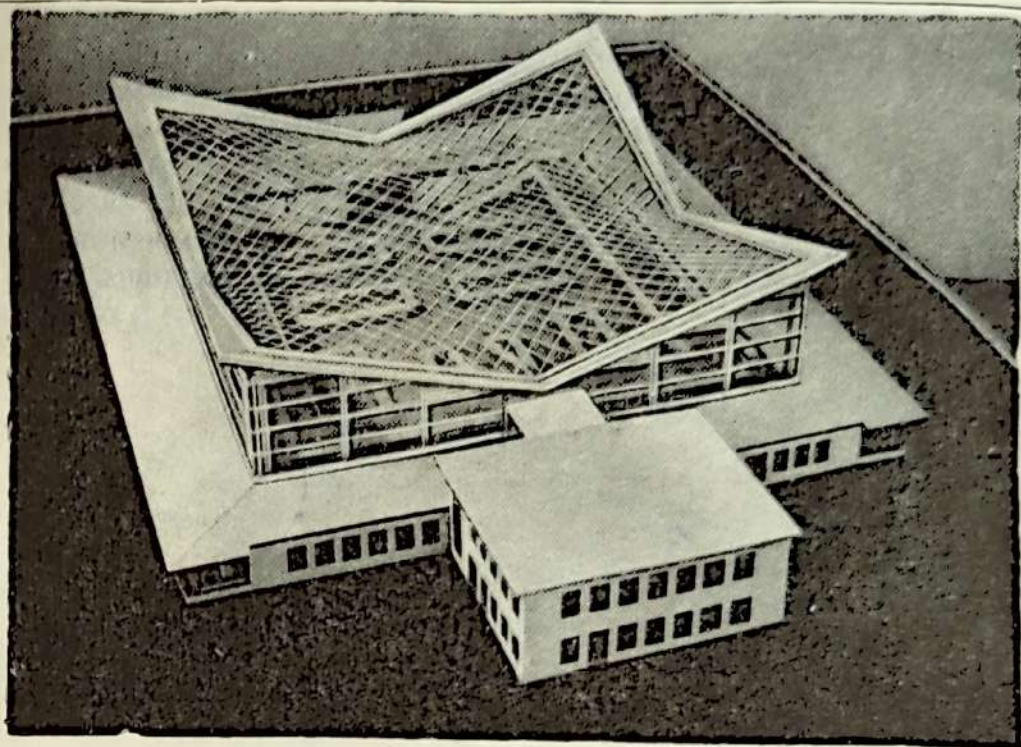


Рис. 41.

в 1963 г. (рис. 42). По наружному контуру покрытие опирается на симметричный плоский контур, круглый в плане, и сплошные стены из сборных железобетонных блоков коробчатого типа, а в центре — на железобетонную опору. Ввиду симметричности контура бортовые элементы легкие, простые по форме и удобные в изготовлении. В плане они образуют шестиугольник, на который в трех направлениях (параллельно главным диагоналям) натягиваются ванты из арматурных стержней. Три центральных ван-

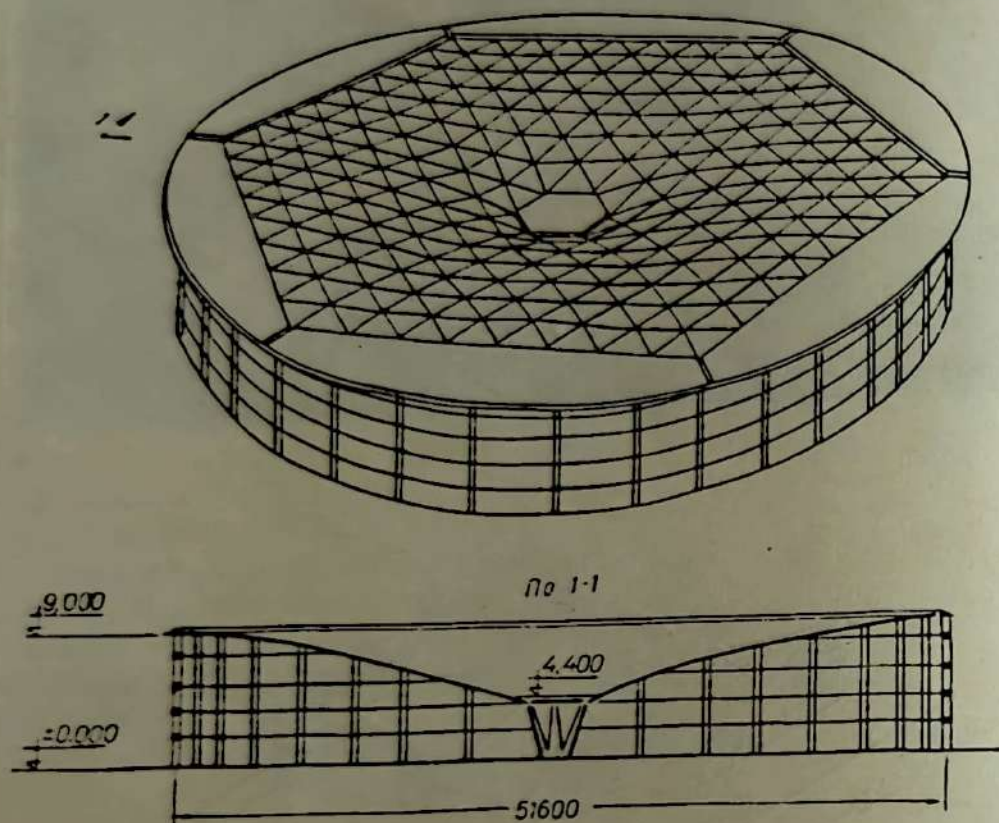


Рис. 42.

та каждого направления прикрепляются к центральной железобетонной опоре, благодаря чему на плоском контуре создается самонапряженная вантовая сеть. Для предварительного натяжения и придания вантовой сети определенной геометрии узлы ее обеспечивают взаимное проскальзывание вантов относительно друг друга. На вантовую сеть навешиваются сборные железобетонные плиты треугольной формы одного типоразмера. Затем швы между ними замоноличиваются.

На первой стадии расчетной схемой покрытия является самонапряженная вантовая сеть с треугольной структурой ячеек, образованная на симметричном плоском шестиугольном контуре и притянутая в центре к железобетонной опоре. Узлы вантовой

сети обеспечивают передачу только вертикальных составляющих от вант. Геометрия вантовой сети определяется расчетом и для заданного контура зависит от принятого предварительного напряжения. Контур вантовой сети представляет собой плоскую шестиугольную раму с жесткими узлами и переменной изгибной и продольной жесткостями стержней. В отношении геометрической изменяемости, различного сечения вант и т. п. расчетная схема такой вантовой сети имеет такие же специфические особенности, как и рассмотренное ранее висячее покрытие.

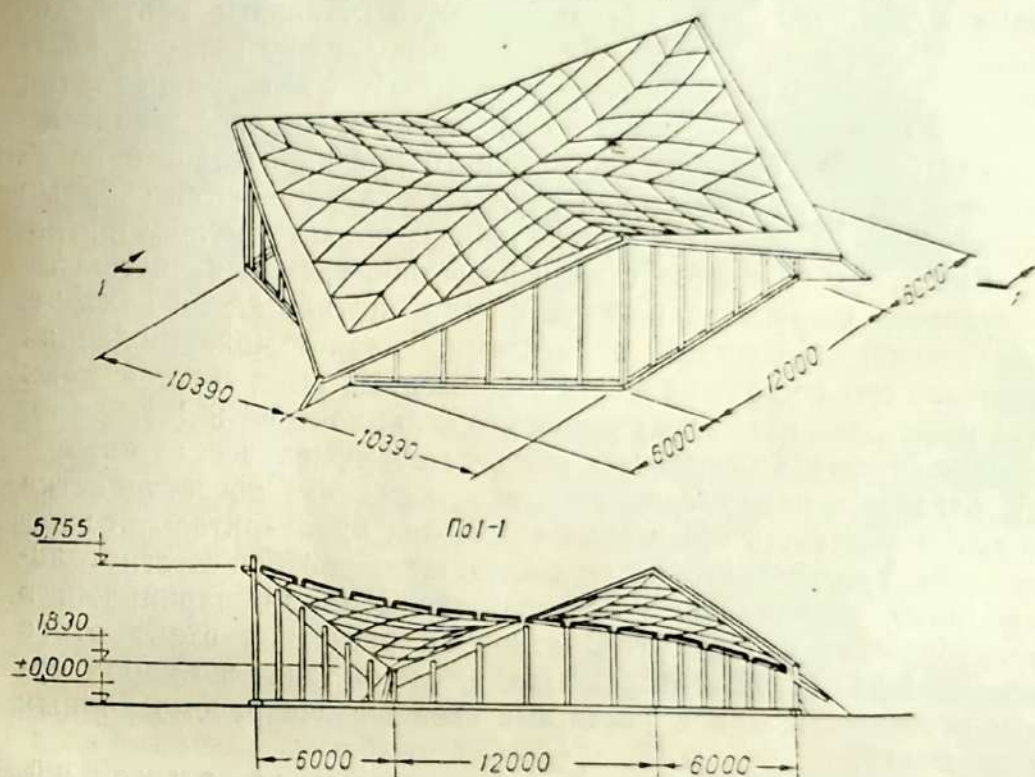


Рис. 43.

На второй стадии расчетной схемой покрытия является железобетонная тонкая пологая оболочка, опирающаяся свободно на наружный контур и центральную опору. Так же, как и ранее рассмотренное висячее покрытие, оболочка имеет треугольную структуру ребер с неодинаковым сечением, а ванты и бортовые элементы оболочки имеют значительные начальные напряжения, соответствующие усилиям при работе сооружения по первой стадии.

Висячее железобетонное покрытие ресторана на 200 мест (рис. 43). Контур его шестиугольный в плане, изломан в пространстве и имеет три приподнятые угловые точки и три опущенные. В верхних точках бортовые элементы соединяются шарнирно, что обеспечивает удобство монтажа на

большой высоте. В трех низких точках соединения бортовых элементов (в местах примыкания к распорным железобетонным опорам) образуется жесткий узел. Кроме трех железобетонных опор, по контуру здания расположены легкие металлические трубчатые опоры, которые одновременно служат импостами остекления. После замоноличивания всех стыков между бортовыми элементами на полученный контур натягивается вантовая сеть из арматурных стержней. Сначала натягиваются три главных диагональных троса, которые одновременно являются тросами жесткости. Затем устанавливаются остальные ванты, которые одним концом прикрепляются неподвижно к тросу жесткости, а вторым — к бортовому элементу. Узел прикрепления ванта к бортовому элементу обеспечивает возможность подтяжки вантов для создания необходимого предварительного натяжения вантовой сети. На вантовую сеть навешиваются сборные железобетонные плиты и замоноличиваются швы между ними.

Видно, что на первой стадии расчетной схемой является нерегулярная косоугольная вантовая сеть, натянутая на контуре, образованном бортовыми элементами. Такая самонапряженная вантовая сеть состоит из шести сопряженных поверхностей, каждая из которых окаймлена одной стороной контура и двумя тросами жесткости. Поверхности соответствует равновесная нагрузка, близкая к равномерно распределенной, чем обеспечивается большая жесткость при малых величинах предварительного натяжения. Контур вантовой сети представляет собой шестиугольную раму, изломанную в пространстве, с тремя шарнирами в трех приподнятых точках. В основном расчетная схема этого покрытия на первой и второй стадиях имеет такие же характерные особенности, как и расчетные схемы выше рассмотренных сооружений.

Тентовые покрытия временных сооружений. Большая экономичность, легкость монтажа, многообразие форм делают эти конструкции весьма эффективными для перекрытия больших пролетов временных летних сооружений. Тентовые покрытия образуются натяжением парусиновой (синтетической) ткани на жесткий или гибкий контур. Ткань тентового покрытия является ограждающей и несущей конструкцией.

Тентовое покрытие летней столовой гостиницы в Каневе (рис. 44).

К стальной стойке, расположенной в центре покрытия, подвешивается наружное кольцо диаметром 15 м и прикрепляется внутреннее кольцо диаметром 2 м. Между этими кольцами, расположенными на разных уровнях, натягивается парусиновый тент. Его полотнище кроится из 24 клиньев.

Расчетной схемой такого сооружения является нитяная сеть, натянутая между двумя разновысокими кольцами. Поверхность такой сети состоит из 24 сопряженных клиньев. Направление

нитей в пределах каждого клина ортогонально, причем основа ткани направлена параллельно апофеме клина. Податливость нитей в двух направлениях различна: податливость по направ-

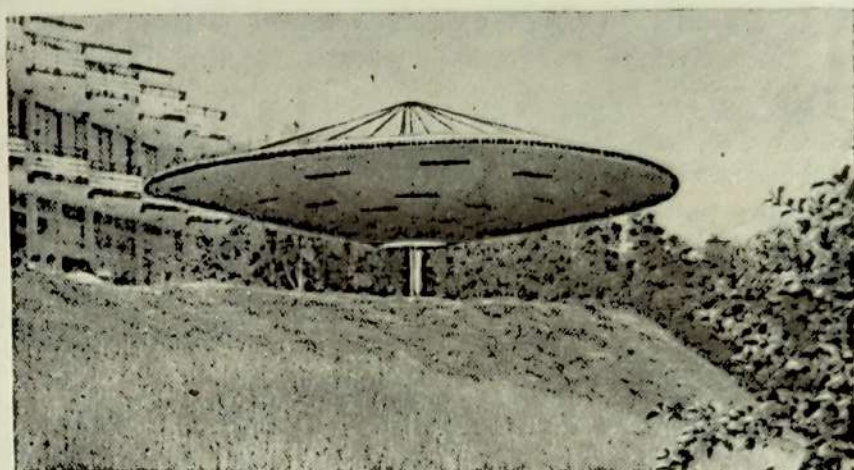


Рис. 44.

лению основы превышает податливость по направлению утка. В местах сопряжения клиньев образуется шов, имеющий большее сечение, чем остальная область ткани. Таким образом структура нитяной сети неоднородна.

Тентовое покрытие летнего кинотеатра (рис. 45). В плане покрытие имеет форму шестиугольника, каждая сторона которого оттягивается пучком арматурных стержней на три высокие и три низкие опоры. Полотнище тента собирается из шести клиньев, форма которых приближается к равностороннему треугольнику. Каждый клин сшивается из стандартных полос. Для создания постоянного натяжения полотнища три высокие опоры запроектированы подвижными. При растяжении

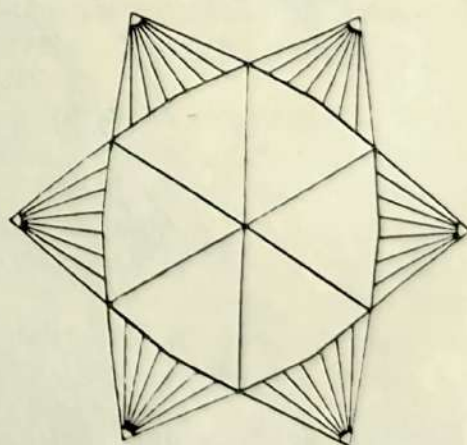
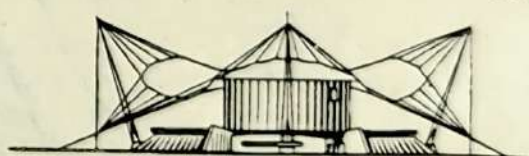


Рис. 45.

ткани они перемещаются, натягивая ткань с постоянной силой.

Расчетной схемой такого сооружения является нитяная сеть, состоящая из шести сопряженных клиньев. Направления нитей основы и утка в пределах каждого клина ортогональны, причем

основа ткани параллельна апофеме треугольника клина. В местах сопряжения клиньев образуется значительная концентрация материала, вызванная забортовкой контурного каната дополнительными накладками и наличием монтажного троса, скрепляющего клинья. В области самого клина в местах швов соединений стандартных полос материала также образуется концентрация материала.

Проект лыжного трамплина Киевского центрального стадиона. Строительство трамплина начато в 1963 г. (рис. 46).

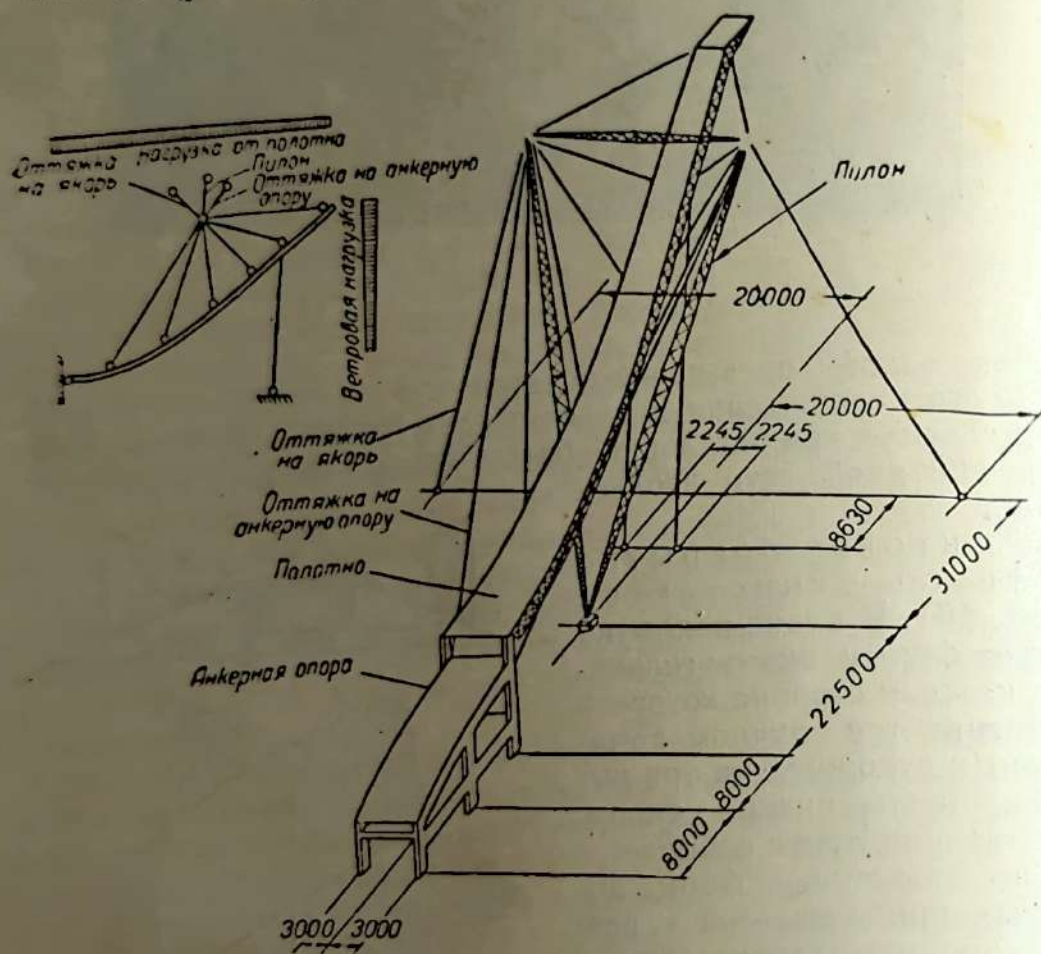


Рис. 46.

Несущие конструкции трамплина запроектированы стальными. Полотно трамплина опирается на стальные фермы и представляют собой жесткую изогнутую полосу, расширяющуюся книзу. Полотно трамплина подвешивается на оттяжках к V-образному пилю. Применение прогрессивной висячей схемы трамплина позволило значительно снизить расход материала по сравнению с существующими подобными сооружениями.

Расчетной схемой трамплина является комбинированная стержневая система, представляющая собой элемент сложной криволинейной формы со значительной изгибной жесткостью (полотно трамплина), подкрепленный нерегулярной стержневой системой. Стержни этой системы имеют различную жесткость.

ПРОБЛЕМА СТАТИЧЕСКОГО РАСЧЕТА ПОЛОГИХ ПРОСТРАНСТВЕННЫХ КОНСТРУКЦИЙ

Широкое применение сборных железобетонных конструкций приводит, как правило, к необходимости рассматривать конструктивные схемы с неравномерно расположенным материалом, концентрируемым в ребрах сборных элементов, в опорных частях сооружения и т. п. Для этих схем характерны анизотропность, нерегулярность структуры. Конструктивные схемы, где основным несущим элементом являются вантовые сети, представляют собой с точки зрения строительной механики стержневых систем геометрически мгновенно изменяемые, а в некоторых случаях просто изменяемые системы. Статический расчет этих конструктивных схем относится к нелинейным задачам строительной механики, связанным с геометрической нелинейностью.

Все это крайне затрудняет практические расчеты с непосредственным применением разработанных методов решения уравнений теории упругости.

Глубоко разработанная теория пластин и оболочек * при применении для практических расчетов, как правило, рассматривает имеющие важное значение отдельные частные случаи, которые приводят к решению в так называемом свернутом виде, либо в рядах. Естественно, что такие решения трудно непосредственно использовать для сложных структур при большом разнообразии граничных условий.

Методы расчета висячих покрытий развиваются на базе достижений науки в области прикладной механики, строительной механики и теории оболочек. Однако общий теоретический уровень в этой области еще не дал возможности создать практически приемлемый и достаточно точный метод расчета таких покрытий. Это объясняется тем, что такие конструкции, являясь в общем случае геометрически мгновенно изменяемыми системами, занимают в строительной механике особое место и привлекли к себе внимание исследователей только в последнее время.

* В разработке теории пластин и оболочек отечественной науке принадлежит ведущая роль.

Подробный обзор и библиография по этому вопросу даны в обобщающих работах В. Г. Коренева и О. Д. Ониашвили, помещенных в сборнике «Строительная механика в СССР» [64].

К первым исследованиям в этой области относятся работы А. Синицына [53], в которых рассматривается расчет сеток как обобщение веревочного многоугольника. А. Синицын основывается на результатах, полученных Г. Маркусом. Приводятся уравнения равновесия в вертикальной плоскости для прямоугольной, радиальной, треугольной и полигональной сетей. При этом натяжение нитей принимается как равнодействующая натяжения мембраны, а не из условия равновесия в плоскости сети. Это приводит к тому, что по длине нити натяжение меняет свою величину. Расчет сеток в работах А. Синицына рассматривается в линейной постановке.

Большое внимание было уделено советскими учеными развитию методов расчета отдельных гибких нитей. Методика расчета отдельных нитей представляет интерес, так как в строительной практике очень часто отдельная нить встречается как самостоятельный конструктивный элемент.

Интересны работы в этой области С. Цаплина [71], Р. Мацелинского [40, 41], В. Качурина [23, 24]. В книге Р. Мацелинского дан вывод основных расчетных формул для гибкой растяжимой нити под любую нагрузку, а также даны оценки погрешностей предложенного метода. В работе В. Качурина [24] рассматриваются также способы статического расчета некоторых пространственных вантовых систем. Основой этих расчетов является теория расчета отдельной гибкой нити. Развивается вопрос расчета нити, имеющей опоры на разных уровнях, влияния смещения опор, а также вопросы расчета жестких нитей.

Некоторым частным случаям решения вантовой сетки посвящены работы Р. Ривлина [74], Х. Банделя [75], Эраса и Эльце [76]. Так, Ривлин рассматривает работу плоской сети, состоящей из двух семейств ортогональных нерастяжимых нитей. При исследовании напряженного состояния сети она заменяется сплошной двумерной средой. В работе Банделя излагается расчет прямоугольной в плане ортогональной нитяной сети — гиперболического параболоида — на вертикальную узловую нагрузку и температурные изменения. Вертикальные перемещения узлов определяются из системы уравнений равновесия сети в деформированном состоянии. Распоры нитей определяются решением методом итерации уравнений, характеризующих растяжимость нитей. Дается также приближенный прием с помощью разложения нагрузки в тригонометрический ряд. В работе Эраса и Эльце предлагается решать обратную задачу, задаваясь системой прогибов для ортогональной сетки отрицательной гауссовой кривизны.

В своей обширной монографии «Висячие покрытия», русский перевод которой вышел под редакцией и с дополнениями И. Людковского, О. Фрей сделал попытку систематизировать различные типы висячих покрытий. Книга носит главным обра-

зом описательный характер и не содержит расчетов и исследований. В 1960 г. в статье Людковского «Подвесные конструкции покрытий в СССР и за рубежом» было дано описание и анализ подвесных конструкций, осуществленных к этому времени.

Расчету вантовых покрытий в 1960—1961 гг. были посвящены статьи И. Глушенко [7]—рассматриваются пологие нити — и Л. Дмитриева [17] — изложен вариационный метод итерационного расчета вантовых сетей. Этот метод является приемлемым для ряда задач строительной практики.

В области висячих покрытий следует отметить научные исследования, проводимые в лабораториях НИИЖБ б. АСИА УССР. К ним относятся работы Людковского [35, 36, 37], в которых дается общий обзор существующих висячих покрытий, классификация и рекомендации по их проектированию. Вопросам расчета отдельных нитей под любую нагрузку посвящены статьи Мацелинского [41]. В работах Лесникова описаны висячие покрытия кругового очертания в плане. В работе Э. Кузнецова [31] приводятся сведения о гибких нитях и вантовых системах. Указывается, что в вантовых системах перемещения можно разделить на упругие и кинематические. Кроме того, высказывается весьма спорное мнение, что в предварительно напряженных (в достаточной степени) системах кинематические перемещения отсутствуют, то есть перемещения типа мгновенной изменемости считаются автором упругими. Рассматриваются изменяемые однопоясные радиальные вантовые системы, при этом внешнее кольцо считается недеформированным, а внутреннее — как один узел. Предлагается метод последовательных приближений, в которых поправка ищется путем решения уравнений, приведенных к линейным. Особый интерес представляет работа Кузнецова [30], в которой формулируется прямая задача инженерного расчета — подбор сечений. Предлагается для нелинейных задач применить метод последовательных деформаций. С математической точки зрения этот метод представляет собой замену нелинейных уравнений рекуррентной последовательностью линейных уравнений. Более подробно метод рассматривается применительно к радиальным вантовым системам.

Идея замены нелинейных уравнений рекуррентной последовательностью линейных лежит в основе метода многоступенчатого нагружения, который разработан в работе Л. Дмитриева [62]. При этом на каждом этапе нагружения рассматривается линейно деформируемая конструкция. Метод разработан с учетом использования электронных вычислительных машин. В работе [19] Л. Дмитриева приведен и другой метод решения задач, имеющих геометрическую нелинейность, в котором при помощи формальных преобразований удается привести нагрузку на вантовую сеть к равновесной, что значительно упрощает решение. Расчет вантовой сети на частные ви-

ды нагрузок (равновесную) рассмотрен в работе В. Гордеева [9]. В работе того же автора [11] приведены уравнения для расчета тканевых оболочек.

Исследованию работы двухпоясных вантовых систем на круглом плане посвящены работы Пшеничникова [45], Кузнецова [30]; Качурина [25] и др., которые рассматривают однопоясную вантовую радиальную систему.

Расчет ортогональной вантовой сети, имеющей поверхность гиперболического параболоида, рассмотрен в работах Битюцкого [2], Загорянского [22], Банделя [75]. В работах Ривлина и Жуковой рассмотрены вопросы расчета плоских вантовых сетей, состоящих из двух ортогональных систем нитей.

В работах Г. Райнуса [48] рассматриваются итерационные методы расчета ферм из тросов, ортогональных сеток из тросов, сеток из тросов и балок на вертикальную нагрузку.

Вопросы экспериментального исследования висячих покрытий рассмотрены в работах Т. Жуковой [21], Х. Лоуля и В. Кульбаха [34] и Е. Харитоновой [70].

Вопросам проектирования висячих конструкций посвящены работы И. Кирсанова [27], Л. Фридмана [69], М. Кольчицкого [28], Л. Дмитриева и А. Городецкого [18].

Вопросам теории и техники расчета сложных статически неопределимых систем уделяется большое внимание. Однако в большинстве случаев общие методы дают теоретическое решение, то есть в конечном счете доказывают только его существование, а для многих важных инженерных задач вообще не существует общих решений. Инженеру необходимо знать, какие напряжения или деформации возникают в конструкции при заданных нагрузках или какие нагрузки можно допустить на данное сооружение. Поэтому большое значение имеют приближенные и численные методы, которые очень часто являются единственно возможными при решении ряда инженерных задач. Точность существующих численных методов является для практики в большинстве случаев вполне достаточной, тем более, что исходные данные обычно являются весьма приближенными. Таким образом, развитие приближенных методов расчета инженерных сооружений, простых и доступных для широкого круга инженеров, приобретает огромное значение. Большое распространение получили численные методы решения дифференциальных уравнений. В этой области следует отметить работы отечественных ученых А. Н. Крылова, Б. Г. Галеркина, Н. М. Крылова, Н. Н. Боголюбова и многих других. В развитии и приложении численных методов интегрирования к различным задачам строительной механики особенно велики заслуги А. Н. Динника и его последователей. Из численных методов за последнее время большое развитие получил метод конечных разностей, или метод сеток. В частности, разностный метод мо-

жет быть с успехом применен для решения весьма актуальных и интересных краевых задач дифференциальных уравнений теории упругости, плоской задачи, задачи кручения, изгиба и т. д. Здесь в первую очередь следует отметить работы П. М. Варвака [3], которые сыграли большую роль во внедрении метода конечных разностей в практику проектирования. Разностный и интерполяционный методы развивали и применяли к различным задачам строительной механики и теории упругости Маркус, Локшин, Корноухов, Галин и др.

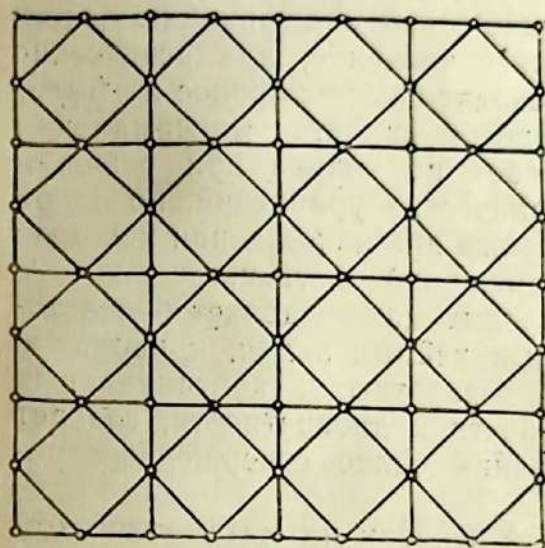


Рис. 47.

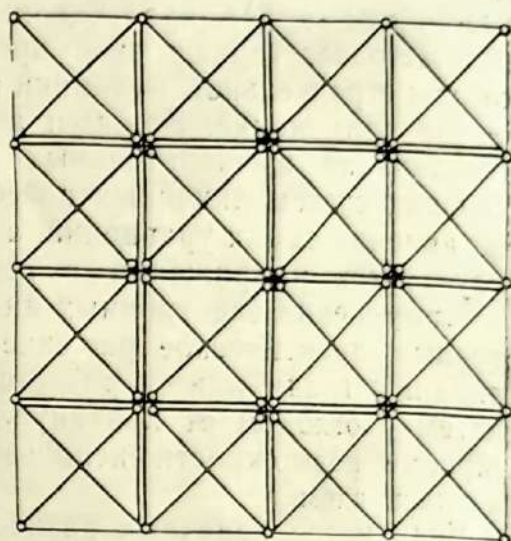


Рис. 48.

В физическом смысле разностные уравнения соответствуют механике дискретной системы, однако в большинстве случаев они применяются для приближенного решения задач механики непрерывной среды. В работе М. И. Длугача [14] показано, что системе уравнений при решении плоской задачи теории упругости по методу сеток соответствует система канонических уравнений метода сил для определенной многократной статически неопределимой стержневой системы (рис. 47).

Можно показать, что конечно-разностным уравнениям изгиба плиты соответствует стержневая система, решаемая методом деформаций (рис. 48).

Численный метод расчета конструкций методом сеток сводится к следующему: действительная конструкция заменяется непрерывной средой с граничными условиями, по возможности соответствующими действительным. Затем для полученной модели составляются дифференциальные уравнения, описывающие ее напряженное состояние. Далее находится приближенное

решение полученных уравнений по методу сеток, соответствующее определению усилий либо перемещений в некоторой стержневой системе.

В последние годы большое значение приобретает численный метод расчета инженерных сооружений, основанный на расчете многократно статически неопределимых систем классическими методами строительной механики стержневых систем. Развитие этого метода тесно связано с применением электронных вычислительных цифровых машин.

Появление вычислительных машин обуславливает более общий подход к задачам строительной механики. При этом частные формы записи, представляющие собой модификацию основных методов, теряют свое прежнее значение, а классические методы строительной механики находят самое широкое и эффективное применение. Важным вопросом является максимальная механизация вычислительных работ не только при решении больших систем линейных алгебраических уравнений, но и при составлении самих уравнений, а также при нахождении искомых результатов напряжений и перемещений и т. п.

Применение электронных цифровых машин делает более доступным практическое решение нелинейных задач, приводит к стиранию грани между расчетом дискретных и континуальных систем с отходом от континуальности к дискретности, создает широкие возможности использования более совершенных расчетных схем.

Если метод конечных разностей основан на математической аппроксимации дифференциальных уравнений, описывающих напряженное состояние конструкции, идеализированной с точки зрения применения самих дифференциальных уравнений, то метод расчета конструкции как многократно статически неопределимой системы основан на идеализации самого сооружения.

Из работ, посвященных разработке и применению этого метода к решению задач строительной механики, необходимо отметить работы отечественных ученых А. П. Филина [68], А. А. Уманского [66] и др., а также работы Аргирова и его последователей [54].

Последние работы связаны с созданием и развитием матричной формы строительной механики с использованием вычислительных машин. При этом для машины программа составляется так, что в ней даны только простейшие указания — матричная инструкция, относящиеся к расположению и порядку матриц. Счет же выполняется с использованием матричной интерпретирующей программы.

Однако матричная форма не является единственно приемлемой и всегда лучшей для расчета сложных сооружений на электронных вычислительных машинах.

Для решения больших по объему задач матричная интерпретирующая программа часто оказывается несостоятельной, так как числовой материал не вмещается в память машины.

Для таких задач необходимы приемы оптимализации программ, основанных на изучении структурных свойств матриц и расчетных схем сооружений.

Программа расчета может быть записана для электрических автоматов в обозначениях инженерного программирования [61], для электронных вычислительных машин—в обозначениях адресного программирования [8] или на языке АЛГОЛ-60 [1].

В настоящей работе уравнения, описывающие статически работу рассматриваемых конструктивных схем, составляются непосредственно моделированием идеализированной схемы сооружения как некоторой пологой стержневой системы.

Форма уравнений, описывающих статическую работу расчетной модели, связана с понятием дискретных операторов, или символов, означающих те или иные алгебраические действия над рассматриваемыми величинами.

При решении нелинейной задачи различается геометрическая нелинейность первого рода и геометрическая нелинейность второго рода.

В зависимости от вида нелинейности в каждом конкретном случае в настоящей работе разработаны способ многоступенчатого нагружения и способ приведения нагрузки к равновесной. Идея способа многоступенчатого нагружения была высказана Б. З. Власовым и в дальнейшем применена В. В. Петровым [43]. Для вантовых сетей способ был разработан автором [62]. Модификация этого способа под названием метода последовательных деформаций была предложена Кузнецовым [29].

Для расчета рассматриваемых конструкций обоими способами составлены программы расчета на клавишных электрических машинах и на электронных цифровых. Программы составлены на международном алгоритмическом языке АЛГОЛ-60. Как показала практика проектирования ряда сооружений, применение таких программ позволило решить многие сложные задачи статического расчета пологих пространственных систем.

ОБОСНОВАНИЕ ПРИНЯТЫХ МЕТОДОВ РАСЧЕТОВ

В практике строительства все большее применение находят конструктивные схемы, при расчете которых необходимо учитывать анизотропность, нерегулярность структуры, упругую работу опорных элементов, нелинейную деформативность.

Это прежде всего характерно для вантовых систем, статическая работа которых меняется по мере их возведения. В процессе монтажа до замоноличивания конструкция вантового покрытия представляет собой комбинированную нелинейно-деформи-

руемую стержневую систему: ванты работают на растяжение, а упругая пространственная рама опорного контура — на изгиб и нормальные силы.

Вантовые покрытия отличаются разнообразием структур секток — прямоугольных, треугольных, шестиугольных, косоугольных и др., регулярность которых нарушается тросами жесткости. Геометрия поверхности лишь в некоторых случаях (например, гиперболический параболоид и его разновидности) может быть представлена в аналитическом виде (выражена с помощью элементарных функций).

После замоноличивания стыков сборных железобетонных плит покрытие превращается в оболочку с ребрами, расположенными по направлениям вант. Таким образом при расчете вантового покрытия рассматривается статическая работа нелинейной комбинированной стержневой системы и ребристой оболочки.

Ребристая оболочка может быть достаточно близко аппроксимирована стержневой системой по следующим соображениям.

Во-первых, основной материал концентрируется в ребрах, ввиду чего конструкция ближе к дискретной системе, чем к континуальной. При рассмотрении ребристой оболочки как стержневой системы заполнение между ребрами заменяется решеткой из стержней и это, очевидно, приводит к меньшим погрешностям, чем по существу произвольное распределение материала ребер по всей конструкции.

Во-вторых, вследствие нерегулярности структуры, анизотропности, сложной геометрии, разнообразия граничных условий чрезвычайно затрудняется запись общих дифференциальных зависимостей теории упругости, предполагающих непрерывность рассматриваемой упругой среды, и применение дифференциальных уравнений для практических расчетов.

В-третьих, при расчетах, производимых с помощью электронных вычислительных машин, дифференциальные уравнения все равно приходится выражать в конечно-разностном виде, что по сути равносильно рассмотрению работы некоторой эквивалентной стержневой системы.

Изложенные соображения, очевидно, могут быть отнесены в равной степени и к другим ребристым оболочкам, не связанным с вантовыми покрытиями, и к плоским ребристым плитам. При таком подходе оказалось возможным составить единый расчетный алгоритм, основанный на рассмотрении пологой пространственной стержневой системы в нелинейной постановке.

Хотя разработанные методы относятся к строительной механике стержневых систем, форма записи уравнений принята другая, характерная для теории упругости, с применением дискретных операторов, что позволило сформулировать основные зависимости наиболее кратко для удобства и общности рассуждений,

обозримости формул. Полученные уравнения равновесия и деформации (более правильно последние назвать уравнениями связи между усилиями и перемещениями) выражают статическую работу в нелинейной постановке нерегулярной стержневой системы с различными жесткостями элементов.

Такая запись позволяет также решать отдельные частные случаи в общем виде и производить необходимые преобразования уравнений (как это делается с дифференциальными уравнениями). Так, например, оказалось возможным при рассмотрении изгиба в вертикальной и горизонтальной плоскостях свести число неизвестных перемещений к трем линейным, исключив углы поворота узлов.

Системы уравнений могут решаться относительно неизвестных перемещений (для линейно-деформируемых систем в этом случае уравнения соответствуют методу деформаций) или относительно неизвестных усилий.

Поскольку общих способов для решения систем нелинейных уравнений не существует, потребовалось разработать приемы линеаризации рассматриваемой системы уравнений. Большинство известных приемов линеаризации основано на применении итерационных процессов. Для данной задачи, как показала практика проектирования, итерационные процессы, в том числе предложенная автором в работе [17] разновидность метода спуска, сходятся медленно и пригодны либо для частных случаев, либо для получения относительно грубого приближения.

В результате проведенного исследования автор остановился на двух способах линеаризации — многоступенчатого нагружения и приведения нагрузки к равновесной.

Геометрическая нелинейность, в которой заключается основная трудность задачи, связана с изменением начальной геометрии в результате деформации системы и присуща всем конструктивным схемам.

Для обычных сооружений при часто встречающихся соотношениях нагрузок и жесткостей деформации настолько малы, что изменением начальной геометрии для распределения усилий в элементах можно пренебречь.

Однако возможно такое соотношение величин нагрузок и жесткости сооружения, когда влияние деформации оказывается существенным. С другой стороны, появление больших деформаций может быть связано с геометрической изменяемостью схемы.

Таким образом геометрическая нелинейность является следствием различных причин и поэтому оказалось целесообразным ввести понятие геометрической нелинейности первого и второго рода. Введение этих понятий облегчило анализ статической работы вантовой сети, на основе которого был предложен способ приведения нагрузки к равновесной.

Каждый из указанных способов имеет свою область рационального применения.

Способ многоступенчатого нагружения благодаря сравнительной простоте алгоритма и цикличности хорошо приспособлен для счета на ЭВМ, но мало пригоден для ручного счета вследствие огромной трудоемкости вычислений.

Способ приведения нагрузки к равновесной менее трудоемок, хотя алгоритм его несколько сложнее и менее универсален, так как пригоден только для систем с геометрической нелинейностью второго рода. Приведенные экспериментальные проверки способа на модели показали хорошее совпадение результатов.

Практика расчета показала, что для задачи средней сложности выполнение расчета способом многоступенчатого нагружения на 10 циклов на клавишных автоматах занимает 20 дней, способом приведения нагрузки к равновесной — 2 дня.

Решение той же задачи, поставленной на ЭВМ БЭСМ-2М с использованием программы для способа многоступенчатого нагружения, потребовало при 10 циклах — 20 сек, при 100 циклах — 3 мин, причем результаты мало отличались. Однако нужно иметь в виду, что подготовка задачи к решению (запись на бланки и перфорирование) занимает 5—6 ч.

Оба способа особенно эффективны для расчета нерегулярных вантовых сетей с большой степенью нелинейности, а также в тех случаях, когда необходим учет горизонтальных смещений.

Как уже упоминалось, разработанный алгоритм охватывает не только вантовые системы, но и другие типы пологих пространственных конструкций. На основе этого алгоритма составлены обобщенные программы на языке АЛГОЛ-60. Программы состоят из отдельных блоков (процедур). Исключив те или иные блоки, можно получить упрощенное решение для различных частных случаев, в том числе и для линейно-деформируемых систем. Кроме того, программа предусматривает упрощения для регулярных систем. Таким образом, формулировка задачи в общем виде не усложняет ее.

Программа для метода многоступенчатого нагружения может быть распространена и на задачи, связанные с общей устойчивостью рассчитываемого сооружения. Так, при решении некоторых задач наблюдалось значительное нарастание прогибов по мере нагружения, что по сути является признаком потери устойчивости.

Введя дополнительные логические операции в программу, можно учесть пластическое перераспределение усилий и в результате оценить общую несущую способность сооружения.

Чрезвычайно важным является не рассмотренный в настоящей работе вопрос динамики конструктивных схем. Вывод соответствующих уравнений, основанных на приведенных зависи-

мостях, не представляет принципиальных трудностей, однако для их решения необходимо дополнительно исследовать возникающие при этом проблемы.

Разработанные программы требуют дальнейшего совершенствования с целью их оптимализации.

Эта работа уже начата (см. гл. IX книги) и продолжается в настоящее время.

Расчеты сооружений показали, что областью рационального применения разработанных методов являются вантовые покрытия, ребристые оболочки и плиты, для которых трудно получить решение в замкнутом виде.

К задачам, для которых уже имеются надежные готовые решения в виде таблиц, формул и т. п., указанную методику применять не имеет смысла.

Глава V

ВЫВОД ОСНОВНЫХ УРАВНЕНИЙ

Инженеру-проектировщику приходится рассчитывать сложные пространственные системы, такие как оболочки, большепролетные плиты, вантовые покрытия и др.

Широкое применение сборных железобетонных конструкций приводит в некоторых случаях к необходимости рассматривать схемы с неравномерно распределенным материалом, концентрируемым в ребрах сборных элементов, в опорных частях сооружения и т. п.

Структура таких схем по характеру статической работы часто бывает близка к структуре нерегулярной стержневой системы. Это во многих случаях затрудняет практические расчеты с непосредственным применением уравнений теории упругости. С другой стороны, развитие электронно-вычислительной техники обусловило необходимость использования для расчетов численных методов, таких как, например, метод сеток, когда дифференциальные уравнения теории упругости заменяются системами алгебраических, конечно-разностных уравнений. Известно, что такая замена по сути означает переход от непрерывной системы с непрерывным, равномерным распределением материала к дискретной системе с концентрацией материала в отдельных стержнях, то есть системы стержневой. Ввиду этого, с нашей точки зрения, при практических расчетах сооружений, составляя уравнения, в ряде случаев более целесообразно исходить не из дифференциальных уравнений теории упругости, а непосредственно моделировать то или иное сооружение как стержневую систему. При таком подходе можно использовать мощный аппарат строительной механики с богатым арсеналом практических приемов. Такой подход тем более уместен для схем с нерегулярной структурой. Однако при составлении расчетной модели сооружения необходимо исходить из уравнений

теории упругости, описывающих напряженное состояние и деформативность отдельных элементов сооружения.

Рассмотрим пологую стержневую систему. Эта система состоит из прямых участков — стержней, соединенных в местах пересечения друг с другом жесткими узлами. Наклон стержней к горизонтальной плоскости (XOY) некоторой прямоугольной системы координат (X, Y, Z) настолько мал, что можно считать синус угла наклона равным его тангенсу.

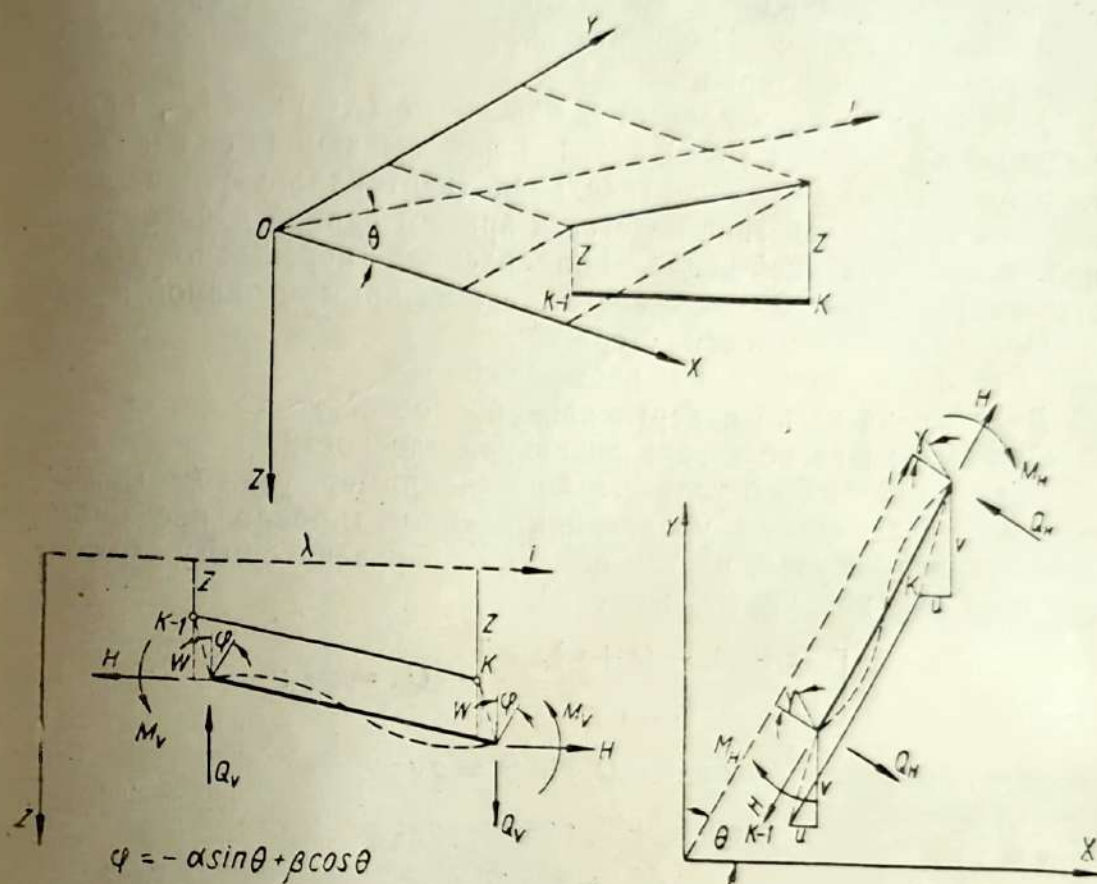


Рис. 49.

Будем также считать, что перемещения узлов в вертикальном направлении в общем случае существенным образом меняют начальную геометрию и изменением ее пренебречь нельзя.

Рассмотрим один из стержней (рис. 49). Пусть некоторый i -тый стержень находится в положении z . Это значит, что концы стержня имеют аппликаты z относительно заданной (левой) системы координат X, Y, Z .

Угол наклона проекции стержня на плоскость (XOY) к оси X обозначим θ . Положительное значение θ — против часовой стрелки вокруг оси.

Приложим к узлам стержневой системы силы q и моменты M . Все узлы получают перемещения u, v, w и повернутся на углы α, β, γ . Положительные значения для линейных перемещений приняты по направлению соответствующих осей X, Y, Z ; для углов поворота — вокруг соответствующих осей по часовой стрелке.

В стержне вследствие деформаций возникнут усилия:

H — горизонтальная составляющая усилия, которую в связи с пологостью стержня примем равной усилию в стержне, направленному по оси. Положительное значение H принимается при растяжении стержня.

Моменты: горизонтальный в плоскости (XOY) — M_n , вертикальный в плоскости (iOZ) — M_v ; перерезывающие силы: Q_n — горизонтальная и Q_v — вертикальная (вертикальную перерезывающую силу вследствие пологости примем равной действительной перерезывающей силе, направленной нормально к оси стержня). Крутильную жесткость стержня примем равной нулю.

Обозначим жесткости стержня:

$A = EF$ — жесткость на растяжение;

$B = EI_v$ — на изгиб в вертикальной плоскости;

$C = EI_n$ — на изгиб в горизонтальной плоскости.

Равновесие любого узла стержня (например, узла k) можно выразить следующими уравнениями, спроектировав все силы, действующие на узел, на три оси X, Y, Z и взяв сумму моментов всех сил вокруг тех же осей.

$$H \left[\frac{(z+w)_k - (z+w)_{k-1}}{\lambda_{k,k-1}} \right] + Q_v = q_z;$$

$$H \cos \theta - Q_n \sin \theta = q_x;$$

$$H \sin \theta + Q_n \cos \theta = q_y;$$

$$M_n = M_z;$$

$$M_v \sin \theta = M_x;$$

$$M_v \cos \theta = -M_y.$$

Для упрощения формы записи применим понятие дискретных операторов для рассматриваемых величин. Для этого прежде всего условимся о направлении стержней. Направление стержней будем ориентировать с помощью оси i , проведенной из начала координат в плоскости (XOY) параллельно проекции стержня на эту плоскость (рис. 50).

Угол наклона θ оси i к оси X отсчитывается против часовой стрелки в пределах $0-180^\circ$.

1) Δ_i — разностный оператор по направлению i , означающий разность двух рядом расположенных величин, деленную на рас-

стояние между ними, а именно: последующая величина минус предыдущая величина в направлении i , деленная на расстояние между ними.

Например, для стержня $k, k-1$ (рис. 51)

$$\Delta z_i = \frac{z_k - z_{k-1}}{\lambda_{k, k-1}}.$$

2) $\Sigma_i []_i^n$ — суммирующий оператор, означающий сумму величин по направлению i в указанных пределах.

Например,

$$\Sigma_i [z]_2^n = z_2 + z_3 + \dots + z_{k-1} + z_k + \dots + z_n;$$

3) $\nabla_k []$ — разностный оператор для узла k , означающий разность величин, зависящую от направлений i , примыкающих

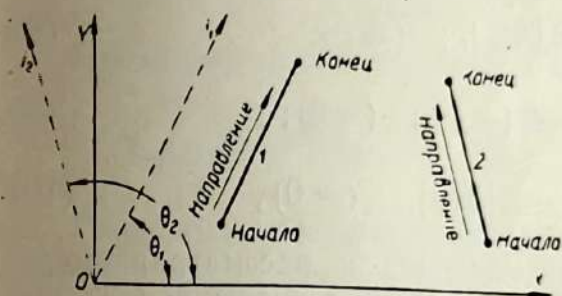


Рис. 50.

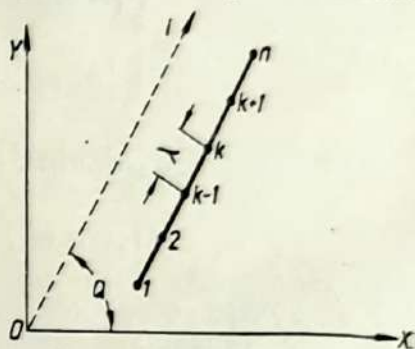


Рис. 51.

к рассматриваемому узлу стержней, то есть величины, относящиеся к стержням, направленным от узла, минус величины, относящиеся к стержням, направленным к узлу.

Например для узла k (рис. 52)

$$\nabla_k [\Delta z] = \Delta z_4 + \Delta z_3 - \Delta z_1 - \Delta z_2;$$

4) Σ_k — суммирующий оператор для узла k , означающий сумму величин, относящихся к примыкающим к рассматриваемому узлу стержням.

Например, для того же узла

$$\Sigma_k [\Delta z] = \Delta z_4 + \Delta z_3 + \Delta z_1 + \Delta z_2.$$

Если в узле сходятся несколько пар стержней, причем каждая пара направлена по одной прямой, разностный оператор для узла при $\lambda = \text{const}$ может быть представлен в следующем виде:

$$\nabla_k [\Delta ()]_i \equiv \Sigma_k [(\Delta^2 [])]_i.$$

Так как

$$\nabla [\Delta z]_i \equiv \sum_{\kappa} [\lambda (\Delta [\Delta z])]_i \equiv \sum_{\kappa} [(\Delta^2 z)]_i,$$

$$\Delta^2 z \equiv \frac{z_{\kappa-1} - 2z_{\kappa} + z_{\kappa+1}}{\lambda}.$$

Перейдя к стержневой системе в целом, для каждого узла (например, того же узла k) соединения стержней, условия равновесия запишем так:

$$\nabla_{\kappa} [H \Delta (z + w) + Q_v]_i = q_z; \quad (5)$$

$$\nabla_{\kappa} [H \cos \theta - Q_H \sin \theta]_i = q_x; \quad (6)$$

$$\nabla_{\kappa} [H \sin \theta + Q_H \cos \theta]_i = q_y; \quad (7)$$

$$\sum_{\kappa} [M_H]_i = \sum_{\kappa} [M_z]_i; \quad (= 0); \quad (8)$$

$$\sum_{\kappa} [M_v \sin \theta]_i = \sum_{\kappa} [M_x]_i; \quad (= 0); \quad (9)$$

$$\sum_{\kappa} [M_v \cos \theta]_i = - \sum_{\kappa} [M_y]_i; \quad (= 0), \quad (10)$$

где i —номера стержней, примыкающих к рассматриваемому узлу; все величины, заключенные в скобки с индексом i , имеют тот же индекс.

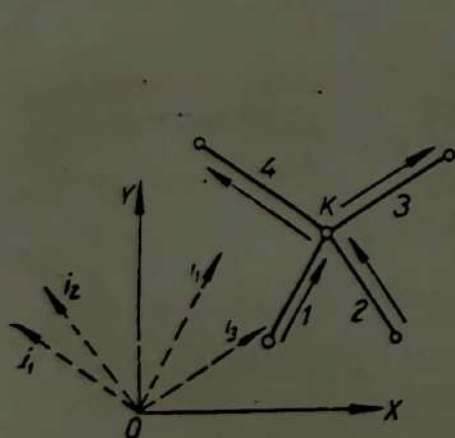


Рис. 52.

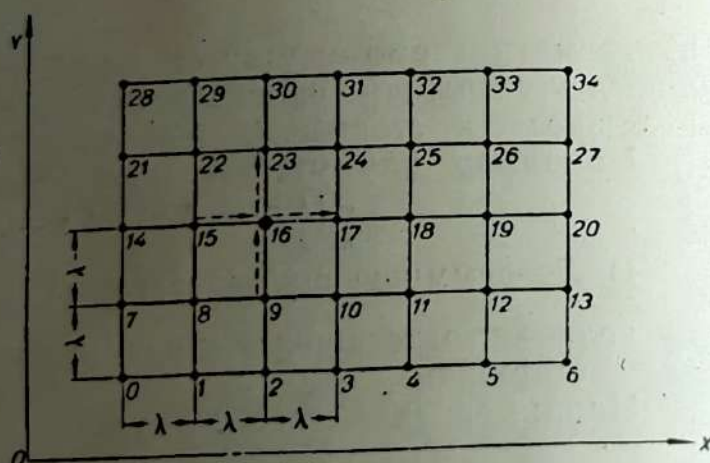


Рис. 53.

Рассмотрим на примере ортогональную плоскую вантовую сеть с начальным натяжением $H = H_0$ и шагом λ , узлы которой нагружены вертикальными силами q_z (рис. 53).

Условия равновесия для каждого узла такой системы согласно (5) запишутся

$$\nabla [H_0 \Delta w]_i = q_z \quad \text{или} \quad \Sigma [H_0 \Delta^2 w] = qz.$$

Напишем уравнение в развернутом виде, например для узла 16,

$$H_0 \left(\frac{w_{23} - w_{16}}{\lambda} + \frac{w_{17} - w_{16}}{\lambda} - \frac{w_{16} - w_9}{\lambda} - \frac{w_{16} - w_{15}}{\lambda} \right) = q_{z, 16},$$

или

$$H_0 \frac{-4w_{16} + w_{15} + w_{23} + w_{17} + w_9}{\lambda} = q_{z, 16}.$$

Рассмотрим деформации стержня.

1. Удлинение.

$$\lambda \Delta l = l_{\text{кон}} - l_{\text{нач}}.$$

Начальная длина стержня

$$l_{\text{нач}} = \sqrt{\lambda^2 + (z_k - z_{k-1})^2} = \lambda \sqrt{1 + \left(\frac{z_k - z_{k-1}}{\lambda} \right)^2}.$$

Разложим корень в степенный ряд, удерживая первые два члена.

$$l_{\text{нач}} = \lambda \left[1 + \frac{1}{2} (\Delta z)^2 \right].$$

После деформации горизонтальная проекция стержня станет равной:

$$\lambda + (u_k - u_{k-1}) \cos \Theta + (v_k - v_{k-1}) \sin \Theta = \lambda [1 + \Delta u \cos \Theta + \Delta v \sin \Theta].$$

По аналогии с определением начальной длины можем написать

$$l_{\text{кон}} = \lambda \left[1 + \frac{1}{2} (\Delta z + \Delta w)^2 \right] \times [1 + \Delta u \cos \Theta + \Delta v \sin \Theta].$$

Тогда удлинение выразится

$$\lambda \Delta l = \lambda \left[\Delta u \cos \Theta + \Delta v \sin \Theta + \Delta w \cdot \Delta z + \frac{1}{2} \Delta w \cdot \Delta w \right].$$

Усилие H выразится

$$H = \frac{\lambda \Delta l}{\lambda} A = A \left[\Delta u \cos \Theta + \Delta v \sin \Theta + \Delta w \cdot \Delta z + \frac{1}{2} \Delta w \cdot \Delta w \right]. \quad (11)$$

2. Изгиб в горизонтальной плоскости.
Стержень перекоится на величину

$$-\Delta u \sin \Theta + \Delta v \cos \Theta.$$

Сечения концов стержня повернутся на угол γ .
Усилия в точке k [46]

$$M_H = \frac{2c}{\lambda} [2\gamma_k + \gamma_{k-1} - 3(\Delta u \sin \Theta - \Delta v \cos \Theta)]; \quad (12)$$

$$Q_H = \frac{6c}{\lambda^2} [\gamma_k + \gamma_{k-1} - 2(\Delta u \sin \Theta - \Delta v \cos \Theta)]. \quad (13)$$

3. Изгиб в вертикальной плоскости.
Стержень перекоится на величину Δw .
Сечения повернутся на угол φ .

$$\varphi = -\alpha \sin \Theta + \beta \cos \Theta.$$

Тогда усилия в точке k определяется [46]:

$$M_v = \frac{2B}{\lambda} [(2\alpha_k + \alpha_{k-1}) \sin \Theta - (2\beta_k + \beta_{k-1}) \cos \Theta - 3\Delta w]; \quad (14)$$

$$Q_v = \frac{6B}{\lambda^2} [(\alpha_k + \alpha_{k-1}) \sin \Theta - (\beta_k + \beta_{k-1}) \cos \Theta - 2\Delta w]. \quad (15)$$

Таким образом уравнения деформаций выглядят так:

$$H = A \left[\Delta u \cos \Theta + \Delta v \sin \Theta + \Delta w \cdot \Delta z + \frac{1}{2} \Delta w \cdot \Delta w \right]; \quad (11)$$

$$M_H = \frac{2c}{\lambda} [2\gamma_k + \gamma_{k-1} - 3(\Delta u \sin \Theta - \Delta v \cos \Theta)]; \quad (12)$$

$$Q_H = \frac{6c}{\lambda^2} [\gamma_k + \gamma_{k-1} - 2(\Delta u \sin \Theta - \Delta v \cos \Theta)]; \quad (13)$$

$$M_v = \frac{2B}{\lambda} [(2\alpha_k + \alpha_{k-1}) \sin \Theta - (2\beta_k + \beta_{k-1}) \cos \Theta - 3\Delta w]; \quad (14)$$

$$Q_v = \frac{6B}{\lambda^2} [(\alpha_k + \alpha_{k-1}) \sin \Theta - (\beta_k + \beta_{k-1}) \cos \Theta - 2\Delta w]. \quad (15)$$

Уравнений равновесия для каждого узла шесть, неизвестных перемещений u , v , w и углов поворота α , β , γ тоже шесть. Таким образом, составив систему алгебраических уравнений для всех узлов и решив их, можно определить все интересующие нас величины.

Уравнения (5—15) можно упростить, сведя число неизвестных к трем перемещениям u , v , w . При этом результаты решения получатся приближенными, а точность решения и структура уравнений будет такой же, как при обычной замене дифференциальных уравнений изгиба конечно-разностными уравнениями.

Для этого достаточно заменить сплошной изгибаемый стержень двумя с переменными сечениями (рис. 54).

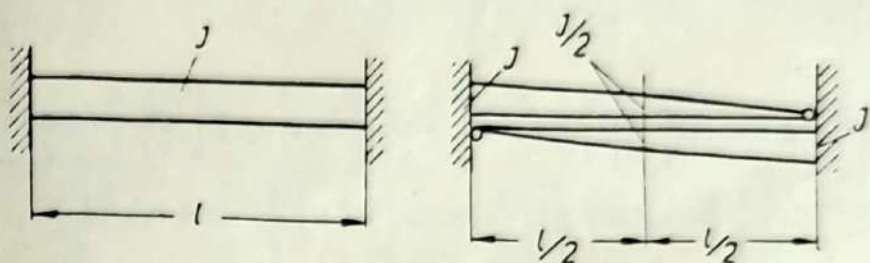


Рис. 54.

Сумма моментов инерции двух стержней в каждом сечении равна моменту инерции исходного стержня.

Усилия от деформаций перекосов и углов поворота такого двойного стержня выразятся:

$$Q_v = \frac{2B}{\lambda^2} [(\alpha_k + \alpha_{k-1}) \sin \Theta - (\beta_k + \beta_{k-1}) \cos \Theta - 2\Delta w]; \quad (16)$$

$$M_v = \frac{2B}{\lambda} [\alpha_k \sin \Theta - \beta_k \cos \Theta - \Delta w]; \quad (17)$$

$$Q_H = \frac{2c}{\lambda^2} [\gamma_k + \gamma_{k-1} - 2(\Delta u \sin \Theta - \Delta v \cos \Theta)]; \quad (18)$$

$$M_H = \frac{2c}{\lambda} [\gamma_k - (\Delta u \sin \Theta - \Delta v \cos \Theta)]. \quad (19)$$

Подставим уравнение (19) в (8) и найдем величину угла поворота γ .

$$\gamma_k \sum_k \left[\frac{c}{\lambda} \right]_i - \sum_k \left[\frac{c}{\lambda} (\Delta u \sin \Theta - \Delta v \cos \Theta) \right]_i = 0.$$

$$\gamma_k = \frac{\sum_k \left[\frac{c}{\lambda} (\Delta u \sin \Theta - \Delta v \cos \Theta) \right]_i}{\sum_k \left[\frac{c}{\lambda} \right]_i}. \quad (20)$$

Уравнение (17) подставим в (9) и (10), откуда найдем α и β .

$$\left. \begin{aligned} \alpha_{\kappa} \sum_{\kappa} \left[\frac{B}{\lambda} \sin^2 \Theta \right]_i - \beta_{\kappa} \sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos \Theta \right]_i &= \sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \sin \Theta \right]_i ; \\ \alpha_{\kappa} \sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos \Theta \right]_i - \beta_{\kappa} \sum_{\kappa} \left[\frac{B}{\lambda} \cos^2 \Theta \right]_i &= \sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \cos \Theta \right]_i ; \end{aligned} \right\}$$

$$\alpha_{\kappa} = \frac{\sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \sin \Theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \cos^2 \Theta \right]_i - \sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \cos \Theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos \Theta \right]_i}{\sum_{\kappa} \left[\frac{B}{\lambda} \sin^2 \Theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \cos^2 \Theta \right]_i - \left(\sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos \Theta \right]_i \right)^2} ;$$

$$\beta_{\kappa} = -$$

$$- \frac{\sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \cos \Theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \sin^2 \Theta \right]_i - \sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \sin \Theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos \Theta \right]_i}{\sum_{\kappa} \left[\frac{B}{\lambda} \cos^2 \Theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \sin^2 \Theta \right]_i - \left(\sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos \Theta \right]_i \right)^2} .$$

Таким образом статическую работу пологой стержневой системы можно выразить следующими уравнениями:

1. Уравнения равновесия

$$\left. \begin{aligned} \nabla [H \Delta (z + w) + Q_v]_i &= q_z ; \\ \nabla [H \cos \Theta - Q_H \sin \Theta]_i &= q_x ; \\ \Delta [H \sin \Theta + Q_H \cos \Theta]_i &= q_y . \end{aligned} \right\}$$

2. Уравнения деформаций

$$\left. \begin{aligned} H &= A \left[\Delta u \cos \Theta + \Delta v \sin \Theta + \Delta w \left(\Delta z + \frac{1}{2} \Delta w \right) \right] ; \\ Q_H &= \frac{2c}{\lambda^2} [\gamma_{\kappa} + \gamma_{\kappa-1} - 2(\Delta u \sin \Theta - \Delta v \cos \Theta)] ; \\ Q_v &= \frac{2B}{\lambda^2} [(\alpha_{\kappa} + \alpha_{\kappa-1}) \sin \Theta - (\beta_{\kappa} + \beta_{\kappa-1}) \cos \Theta - 2\Delta w] , \end{aligned} \right\}$$

где

$$\alpha_{\kappa} = \frac{\sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \sin \theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \cos \theta \right]_i - \sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \cos \theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \sin \theta \cos \theta \right]_i}{\sum_{\kappa} \left[\frac{B}{\lambda} \sin^2 \theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \cos^2 \theta \right]_i - \left(\sum_{\kappa} \left[\frac{B}{\lambda} \sin \theta \cos \theta \right]_i \right)^2}; \quad (21)$$

$$\beta_{\kappa} = -$$

$$- \frac{\sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \cos \theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \sin^2 \theta \right]_i - \sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \sin \theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \sin \theta \cos \theta \right]_i}{\sum_{\kappa} \left[\frac{B}{\lambda} \sin \theta \right]_i \sum_{\kappa} \left[\frac{B}{\lambda} \cos^2 \theta \right]_i - \left(\sum_{\kappa} \left[\frac{B}{\lambda} \sin \theta \cos \theta \right]_i \right)^2}; \quad (22)$$

$$\gamma_{\kappa} = \frac{\sum_{\kappa} \left[\frac{c}{\lambda} (\Delta u \sin \theta - \Delta v \cos \theta) \right]_i}{\sum_{\kappa} \left[\frac{c}{\lambda} \right]_i}.$$

Рассмотрим случаи закрепления узлов от смещений, часто встречающиеся при различных схемах сооружений:

1) наличие в узле трех связей, как угодно направленных (разумеется при условии, что каждая из них не совпадает по направлению с любой другой и не находится в одной плоскости), делает невозможным перемещение узла (рис. 55). В этом случае все три составляющие перемещения равны нулю:

$$u=0; v=0; w=0;$$

2) в узле имеется связь, совпадающая по направлению с какой-либо составляющей перемещения. Соответствующее перемещение принимается равным нулю (рис. 56);

3) узел закреплен связью, наклоненной к оси z под некоторым углом φ , проекция которой на плоскость (XOY) имеет с осью X угол ψ (рис. 57).

Поскольку перемещение узла возможно только в плоскости, нормальной к оси связи, вертикальная составляющая перемещения w связана с горизонтальными составляющими u и v следующей зависимостью:

$$w = (u \cos \psi + v \sin \psi) \operatorname{tg} \varphi;$$

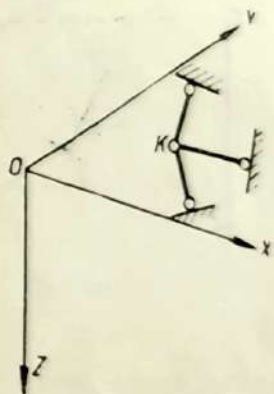


Рис. 55.

4) узел закреплен связью в плоскости (XOY), направленной к оси X под углом ψ (рис. 58).

Горизонтальные составляющие перемещений в этом случае связаны зависимостью

$$u \cos \psi + v \sin \psi = 0;$$

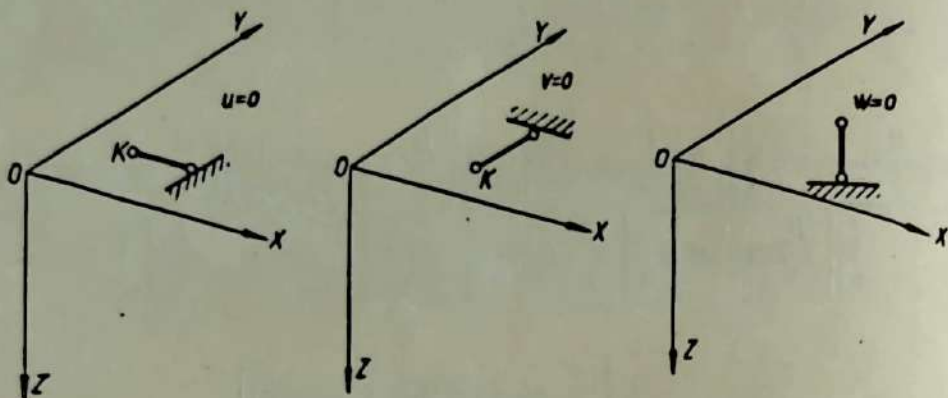


Рис. 56.

5) при наличии в узле защемлений, препятствующих повороту вокруг осей X , Y , Z , соответствующие углы поворота принимаются равными нулю.

$$\alpha = 0; \beta = 0; \gamma = 0;$$

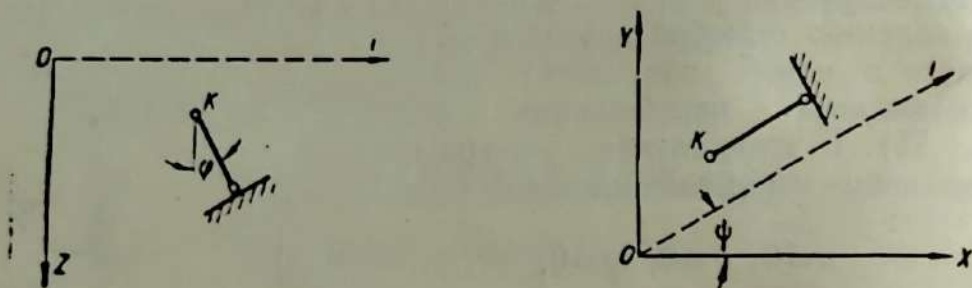


Рис. 57.

6) узел закреплен от поворота в вертикальной плоскости, причем плоскость защемления направлена под углом ψ к оси X (рис. 59). Величины углов поворота можно найти из условий равенства нулю суммы моментов, примыкающих к узлу стержней вокруг линии пересечения плоскости защемления с плоскостью (XOY), и условия равенства нулю поворота узла вокруг оси, нормальной к плоскости защемления.

$$\sum_k [M_{\sigma} \cos (\psi - \theta)]_i = 0;$$

$$\alpha \sin \psi - \beta \cos \psi = 0.$$

Подставив значения моментов, получим

$$\begin{aligned} \alpha_{\kappa} &= \sum_{\kappa} \left[\frac{2B}{\lambda} \sin \Theta \cos (\psi - \Theta) \right]_i - \beta_{\kappa} \sum_{\kappa} \left[\frac{2B}{\lambda} \cos \Theta \cos (\psi - \Theta) \right]_i = \\ &= \sum_{\kappa} \left[\frac{2B}{\lambda} \Delta w \cos (\psi - \Theta) \right]_i ; \\ \alpha_{\kappa} \sin \psi - \beta_{\kappa} \cos \psi &= 0; \end{aligned}$$

откуда

$$\begin{aligned} \alpha_{\kappa} &= \frac{\sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \cos (\psi - \Theta) \right]_i}{\sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos (\psi - \Theta) \right]_i - \operatorname{tg} \psi \sum_{\kappa} \left[\frac{B}{\lambda} \cos \Theta \cos (\psi - \Theta) \right]_i} ; \\ \beta_{\kappa} &= \frac{\sum_{\kappa} \left[\frac{B}{\lambda} \Delta w \cos (\psi - \Theta) \right]_i}{\operatorname{ctg} \psi \sum_{\kappa} \left[\frac{B}{\lambda} \sin \Theta \cos (\psi - \Theta) \right]_i - \sum_{\kappa} \left[\frac{B}{\lambda} \cos \Theta \cos (\psi - \Theta) \right]_i} . \end{aligned}$$

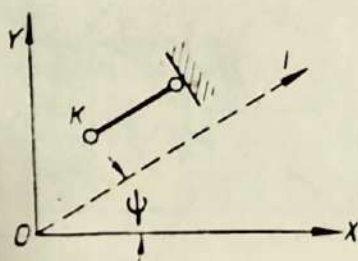


Рис. 58.

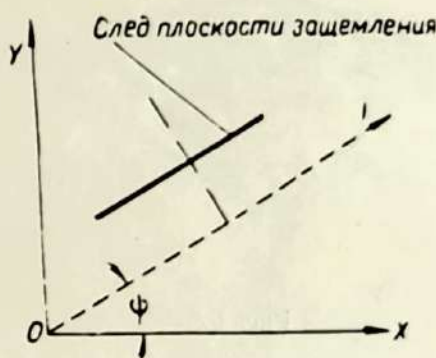


Рис. 59.

Весьма часто рассматриваемая система имеет линии симметрии. В том случае, если нагрузка также симметрична, отдельные части стержневой системы работают одинаково, то есть в них возникают одни и те же усилия, узлы получают одинаковое перемещение. Поэтому для расчета достаточно рассмотреть одну из частей системы.

По обе стороны линий симметрии усилия одинаковы, поэтому перемещения, нормальные к линиям симметрии, равны нулю. Это равносильно наложению соответствующих связей.

При рассмотрении симметричной задачи из всей стержневой системы выделяется часть, отсекаемая линиями симметрии, и

на все узлы, находящиеся на этих линиях, накладываются горизонтальные связи и плоскости защемления, направленные нормально к линиям симметрии. К узлам, находящимся на линиях симметрии, прикладываются половинные нагрузки.

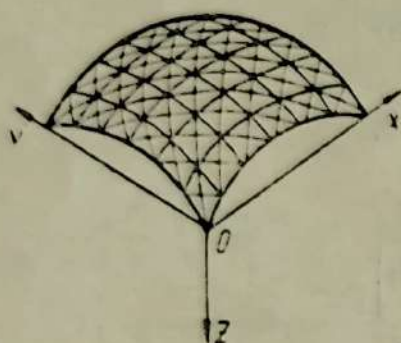


Рис. 60.

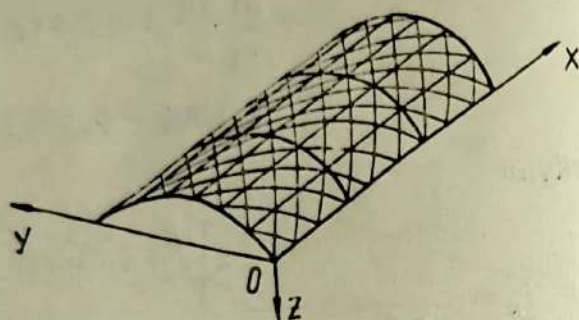


Рис. 61.

Рассмотрим различные конструктивные схемы как частные случаи стержневой системы.

1. Ребристая оболочка (в линейной постановке) (рис. 60, 61, 62).

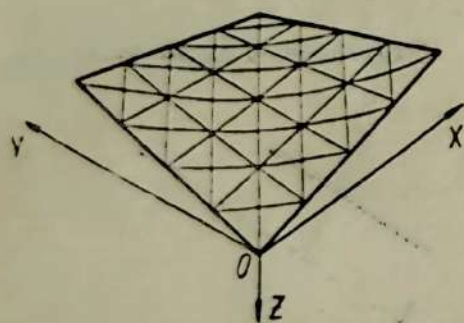


Рис. 62.

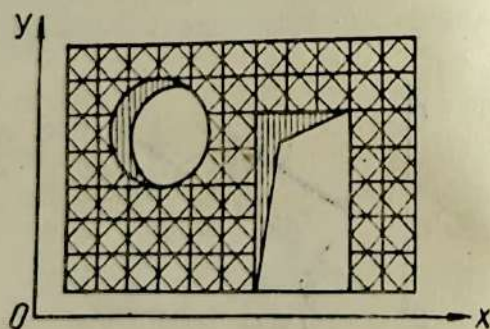


Рис. 63.

Уравнения равновесия:

$$\nabla [H \Delta z + Q_v]_i = q_z;$$

$$\nabla [H \cos \theta]_i = q_x;$$

$$\nabla [H \sin \theta]_i = q_y.$$

Уравнения деформаций:

$$H = A[\Delta u \cos \theta + \Delta v \sin \theta + \Delta w \cdot \Delta z];$$

$$Q_v = \frac{2B}{\lambda^2} [(x_k + x_{k-1}) \sin \theta - (\beta_k + \beta_{k-1}) \cos \theta - 2\Delta w].$$

2. Ребристая панель—балка-стенка (рис. 61).

Уравнения равновесия:

$$\nabla [H \cos \Theta]_i = q_x;$$

$$\nabla [H \sin \Theta]_i = -q_y.$$

Уравнение деформации

$$H = A[\Delta u \cos \Theta + \Delta v \sin \Theta].$$

3. Ребристая плита (рис. 64, 65).

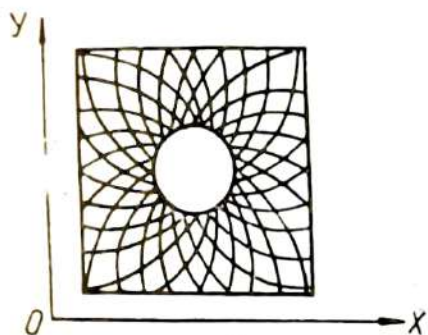


Рис. 64.

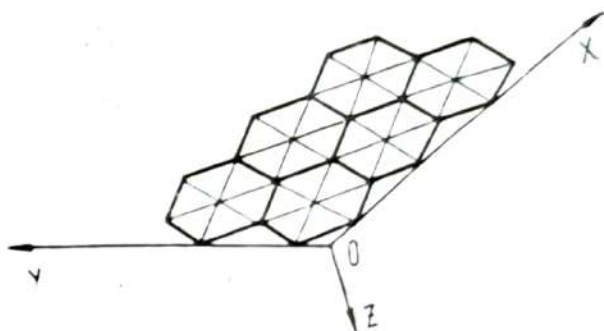


Рис. 65.

Уравнение равновесия

$$\nabla [Q_v]_i = q_z.$$

Уравнение деформации

$$Q_v = \frac{2B}{\lambda^2} [(\alpha_k + \alpha_{k-1}) \sin \Theta - (\beta_k + \beta_{k-1}) \cos \Theta - 2\Delta w].$$

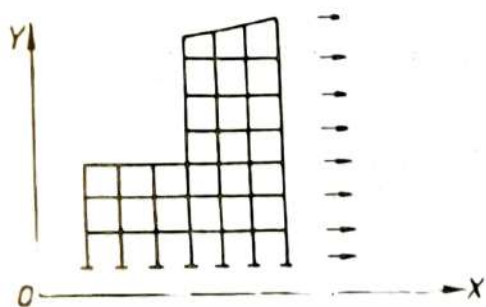


Рис. 66.

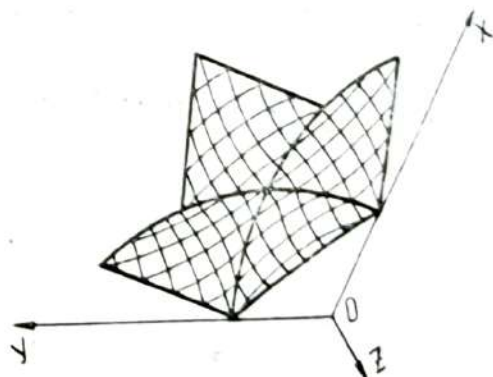


Рис. 67.

4. Высокий многоярусный, рамный каркас (приближенное решение на горизонтальную нагрузку, рис. 66).

Уравнение равновесия

$$\nabla [-Q_k \sin \Theta]_i = q_x.$$

Точное решение выражено уравнениями (6), (7), (8), (12), (13).

Уравнение деформации

$$Q = \frac{2c}{\lambda^2} [\gamma_k + \gamma_{k-1} - 2\Delta u \sin \Theta].$$

5. Безмоментная оболочка (вантовая сеть в нелинейной постановке, рис. 67).

Уравнения равновесия:

$$\nabla [H \Delta (z + w)]_i = q_z;$$

$$\nabla [H \cos \Theta]_i = 0;$$

$$\nabla [H \sin \Theta]_i = 0.$$

Уравнение деформации

$$H = H_0 + A \left[\Delta u \cos \Theta + \Delta v \sin \Theta + \Delta w \cdot \Delta z + \frac{1}{2} (\Delta w)^2 \right],$$

где H_0 —величина предварительного натяжения.

Введем понятие нити. Нитью назовем плоскую ломаную линию, образованную стержнями. Во всех узлах, кроме крайних, пересекаются не более двух нитей. Таким образом рассматриваемую стержневую систему можно образовать из отдельных нитей.

Характерным для нити является постоянство распора по ее длине. Используя это обстоятельство, упростим уравнение деформации. Просуммируем распоры всех стержней нити.

$$\begin{aligned} \sum_i |H_i|^m = A \left[\sum_i |\Delta u \cos \Theta_i|^m + \sum_i |\Delta v \sin \Theta_i|^m + \sum_i |\Delta w \cdot \Delta z_i|^m + \right. \\ \left. + \frac{1}{2} \sum_i |\Delta w \cdot \Delta w_i|^m \right] + \sum |H_0|^m, \end{aligned}$$

где i —номер стержня нити;

m —число стержней в нити.

$$\begin{aligned} H = H_0 + A \left[\Delta^* u \cos \Theta + \Delta^* v \sin \Theta + \frac{1}{m} \sum_i |\Delta w \cdot \Delta z_i|^m + \right. \\ \left. + \frac{1}{2m} \sum_i |(\Delta w^2)_i|^m \right], \end{aligned}$$

так как $\frac{1}{m} \sum_i |\Delta u_i|^m = \frac{u_{\text{нач}} - u_{\text{кон}}}{\lambda \cdot m} \equiv \Delta^* u.$

Таким образом уравнения принимают следующий вид.
Уравнения равновесия:

$$\nabla [H \Delta (z + w)]_i = q_z; \quad (23)$$

$$\nabla [H \cos \theta]_i = 0; \quad (24)$$

$$\nabla [H \sin \theta]_i = 0. \quad (25)$$

Уравнение деформаций

$$H = H_0 + A \left[\Delta^x u \cos \theta + \Delta^x v \sin \theta + \frac{1}{m} \sum_i |\Delta w \cdot \Delta z|_1^m + \right. \\ \left. + \frac{1}{2m} \sum_i |(\Delta w)^2|_1^m \right], \quad (26)$$

где $\Delta^x u \equiv \frac{u_{\text{кон}} - u_{\text{нач}}}{\lambda \cdot m}$ — конечное значение минус начальное, деленное на расстояние между ними.

Уравнений (23) столько, сколько узлов в вантовой сети; уравнений (24) и (25) столько, сколько узлов сопряжений начальных и конечных точек нитей; уравнений (26) столько, сколько нитей.

В случае податливого в горизонтальной плоскости изгибаемого контура для контурных точек добавляются уравнения равновесия

$$\nabla [-Q_H \sin \theta]_i = 0;$$

$$\nabla [Q_H \cos \theta]_i = 0$$

и уравнение деформации

$$Q_H = \frac{2B}{\lambda^2} [\gamma_k + \gamma_{k-1} - 2(\Delta u \sin \theta - \Delta v \cos \theta)].$$

В случае изгибания бортового элемента в двух плоскостях (горизонтальной и вертикальной) для контурных точек должны быть составлены три уравнения равновесия (5), (6), (7) и три уравнения деформаций (11), (16), (18).

6. Плоская стержневая система.

Уравнения равновесия:

$$\nabla [H \cos \theta - Q_H \sin \theta]_i = q_x;$$

$$\nabla [H \sin \theta + Q_H \cos \theta]_i = q_y.$$

Уравнения деформаций:

$$H = A[\Delta u \cos \theta + \Delta v \sin \theta];$$

$$Q_H = \frac{2B}{\lambda^2} [\gamma_K + \gamma_{K-1} - 2(\Delta u \sin \theta - \Delta v \cos \theta)].$$

Таким образом, полученная система уравнений описывает статическую работу достаточно широкого круга конструктивных систем, даже таких, как оболочки и комбинированные стержневые системы.

Глава VI

РАСЧЕТ ПОЛОГИХ ПРОСТРАНСТВЕННЫХ СИСТЕМ В НЕЛИНЕЙНОЙ ПОСТАНОВКЕ (ВАНТОВЫЕ СЕТИ)

По сравнению с другими пологими пространственными системами нелинейность наиболее сильно выражена в вантовых покрытиях. Поэтому для решения нелинейной задачи рассматриваются вантовые сети, хотя все сказанное ниже в большей или меньшей степени относится и к остальным системам.

Вантовые сети, являющиеся основным несущим элементом вантовых покрытий, как правило, представляют собой, с точки зрения строительной механики стержневых систем, геометрически мгновенно изменяемые, а в некоторых случаях просто изменяемые системы. Статическая работа вантовой сети описывается уравнениями (23—26).

Уравнения равновесия:

$$\nabla [H \Delta(z + w)]_i = q_z;$$

$$\nabla [H \cos \Theta]_i = q_x;$$

$$\nabla [H \sin \Theta]_i = q_y$$

Уравнение деформации

$$H = H_0 + A \left[\frac{1}{m} \sum_i |\Delta w \cdot \Delta z|_1^m + \frac{1}{2m} \sum |\Delta w \cdot \Delta w|_1^m + \Delta^x u \cos \Theta + \right. \\ \left. + \Delta^x v \sin \Theta \right].$$

Эти уравнения нелинейны относительно искомых перемещений u , v , w , поскольку последние входят в уравнения не только в первой, но и во второй и третьей степенях.

Нелинейность эта геометрическая. При приложении нагрузки к вантовому покрытию в нитях, образующих несущую ванто-

вую сеть, возникают дополнительные усилия. Эти усилия связаны с деформацией вантового покрытия — с перемещениями u , v , w . Легко заметить из уравнения (26), что эти величины пропорциональны перекосам стержней, образующих нить, и их квадратам (Δw , Δu , Δv — есть не что иное, как перекосы стержней).

Вследствие этого по мере возрастания приложенной нагрузки отдельные части уравнения (26), из которых складывается величина дополнительного усилия в нити, будут изменяться в разной степени.

Пусть для некоторой нагрузки q_z , q_x , q_y известно решение, то есть найдены перемещения ($w=z_1$; $u=u_1$; $v=v_1$), соответствующие этой нагрузке.

При изменении нагрузки q_z , q_x , q_y на бесконечно малые величины dq_z , dq_x , dq_y перемещения также изменятся на бесконечно малую величину du , dv , dw , а величины дополнительных усилий — соответственно на dH .

Уравнения (23—26) примут вид:

$$\nabla [H \Delta (z + z_1 + dw) + dH \cdot \Delta (z + z_1 + dw)]_i = q_z + dq_z;$$

$$\nabla [(H + dH) \cos \Theta]_i = q_x + dq_x;$$

$$\nabla [(H + dH) \sin \Theta]_i = q_y + dq_y;$$

$$H + dH = H_0 + A \left[\frac{1}{m} \sum_i [\Delta (z_1 + dw) \Delta z]_i^m + \right. \\ \left. + \frac{1}{2m} \sum_i [(\Delta (z_1 + dw))^2]_i^m + \Delta (u_1 + du) \cos \Theta + \Delta (v_1 + dv) \sin \Theta \right].$$

При пренебрежении квадратами и кубами приращений перемещений и усилий как величинами высших порядков малости уравнения примут вид:

$$\nabla [H \Delta (z + z_1 + dw) + dH \Delta (z + z_1)]_i = q_z + dq_z;$$

$$\nabla [(H + dH) \cos \Theta] = q_x + dq_x;$$

$$\nabla [(H + dH) \sin \Theta] = q_y + dq_y;$$

$$H + dH = H_0 + A \left[\frac{1}{m} \sum_i |\Delta z_1 \Delta z|_i^m + \frac{1}{2m} \sum_i |\Delta z_1 \Delta z_1|_i^m + \right.$$

$$+ \frac{1}{m} \sum_i |\Delta (z + z_1) \Delta du|_i^m + \Delta^x u_1 \cos \Theta + \Delta^x v_1 \sin \Theta +$$

$$\left. + \Delta^x du \cos \Theta + \Delta^x dv \sin \Theta \right].$$

В результате сокращения в правых и левых частях уравнений равных выражений система будет выглядеть следующим образом:

$$\nabla [H \Delta dw + dH \Delta (z + z_1)]_i = dq_z; \quad (27)$$

$$\nabla [dH \cos \Theta]_i = dq_x; \quad (28)$$

$$\nabla [dH \sin \Theta]_i = dq_y; \quad (29)$$

$$dH = A \left[\frac{1}{m} \sum_i |\Delta (z + z_1) \Delta dw|_1^m + \Delta^x du \cos \Theta + \Delta^x dv \sin \Theta \right]. \quad (30)$$

Получена линейная система уравнений.

Для частного случая, когда вантовая сеть образована нитями, неподвижно закрепленными на контуре, уравнения (27), (28), (29) имеют вид:

$$\sum_{\kappa} [H \Delta^2 (dw) + dH \Delta^2 (z + z_1)]_i = dq_z; \quad (31)$$

$$dH = A \frac{1}{m \lambda} \sum_i [\Delta^2 (z + z_1) dw]_1^m; \quad (32)$$

$$H = H_0 + A \frac{1}{m \lambda} \sum_i [\Delta^2 z \cdot z_1 + 0,5 \Delta^2 z_1 \cdot z_1]_1^m. \quad (33)$$

За исходное положение может быть принято любое, в том числе ненагруженное состояние вантовой сети, когда нагрузка $q_x = q_y = q_z = 0$, а следовательно $z_1 = 0$ и $H = H_0$.

Таким образом, решение системы нелинейных уравнений может быть получено с точностью до величин высших порядков малости, если прикладывать нагрузку бесконечно малыми порциями, считая каждое предыдущее состояние за исходное и решая каждый раз систему линейных уравнений. Однако для практических расчетов можно загружать вантовое покрытие не бесконечно малой, а достаточно малой нагрузкой конечной величины, как это делается в известном способе конечных разностей, когда бесконечно малые величины в дифференциальных уравнениях заменяются малыми конечными величинами. Тогда решение системы нелинейных уравнений сводится к повторному решению систем линейных с меняющимися коэффициентами.

При расчете обычных строительных конструкций принимается, что перемещения узлов невелики, и на этом основании конструкции считаются линейно-деформируемыми. Статическая работа их описывается линейными уравнениями. Такое допущение равносильно предположению, что начальная геометрия системы после загрузки не изменилась существенным образом, то есть, что начальная и конечная геометрические формы системы практически не отличаются друг от друга вследствие того, что перемещения узлов — величины очень малые.

Однако, если узлы рассматриваемой системы перемещаются вследствие тех или иных причин на значительные расстояния, то указанное допущение приведет к значительным погрешностям при определении напряженного состояния и деформативности системы, так как окончательная геометрия системы в этом случае будет существенным образом отличаться от начальной, что в итоге скажется на распределении усилий в элементах системы.

Введем понятия геометрической нелинейности первого и второго рода. Для этого рассмотрим возможные причины возникновения больших перемещений.

Прежде всего нагрузка, приложенная к узлам системы, может быть настолько большой, что вызовет значительные деформации системы. Большие перемещения возможны также при потере устойчивости системы, когда в результате изменения начальной геометрии под действием нагрузки больше критической усилия в элементах системы возрастают до значений, при которых узлы перемещаются на достаточно большие расстояния.

Нелинейность, связанную с количественной характеристикой нагрузки, назовем геометрической нелинейностью первого рода.

Другой причиной возникновения больших перемещений может быть геометрическая изменяемость системы. Каждая нить вантовой сети представляет собой геометрически изменяемую шарнирную цепь, которая для восприятия той или иной нагрузки должна принять соответствующую этой нагрузке форму, то есть изменить существенным образом свою начальную геометрию.

Насколько сильно изменится начальная форма нити, зависит от характера нагрузки, от соотношения величин узловых нагрузок. Для каждой нити в зависимости от ее начальной формы существует один вид нагрузки, восприятие которой возможно без существенного изменения начальной геометрии.

Такая нагрузка (в дальнейшем называемая равновесной) должна быть пропорциональна начальным кривизнам нити или при дискретном приложении нагрузки пропорциональна величинам $\nabla [\Delta z]_i$.

Поскольку в этом случае рассматривается отдельная нить, когда к узлу примыкают всего два стержня; величина $\nabla [\Delta z]_i$ при $\lambda = \text{const}$ распишется

$$\nabla [\Delta z] \equiv \frac{z_{k+1} - z_k}{\lambda_{k,k+1}} - \frac{z_k - z_{k-1}}{\lambda_{k,k-1}} \equiv \Delta^2 z.$$

Для вантовой сети в целом равновесная нагрузка представляет собой линейную комбинацию равновесных нагрузок отдельных нитей.

$$q_{\text{равн}} = \nabla [c \Delta z]_i,$$

где c — коэффициент пропорциональности.

Чем больше нагрузка по своему характеру отличается от равновесной, тем значительнее отличие конечной формы нитей от начальной и, следовательно, тем сильнее будет сказываться нелинейность системы. Нелинейность, связанную с качественной характеристикой нагрузки, назовем геометрической нелинейностью второго рода.

Например, пусть некоторая нить имеет начальную форму, описанную по параболе, что означает расположение узлов нити на кривой, описываемой уравнением параболы.

Тогда при $\lambda = \text{const}$ $\nabla [\Delta z]_i = \text{const}$. Если эту нить загрузить равномерной нагрузкой (рис. 68), то есть силами, одинаковыми по величине $q_z = \text{const}$, то вследствие упругого удлинения нить займет новое положение по форме, подобной начальной, то есть будет описываться по параболе $\nabla [\Delta(z+w)] = \text{const}$. Величина прогибов будет зависеть от величины нагрузки. При некоторых больших значениях нагрузки прогибы могут быть большими.

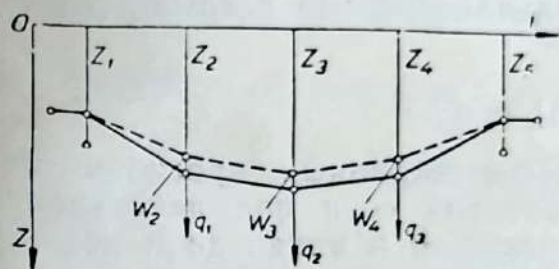


Рис. 68.

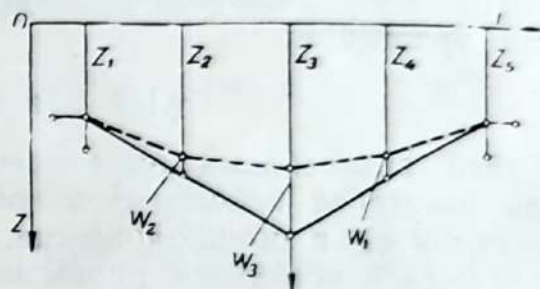


Рис. 69.

В этом случае будет иметь место нелинейность первого рода. При обычных (часто встречающихся) значениях нагрузки и упругих характеристик нитей нелинейностью первого рода можно пренебречь.

Однако, если ту же нить загрузить иной по характеру нагрузкой, например одной силой (рис. 69), то для восприятия такой нагрузки нить существенным образом изменит свою начальную форму, прогнувшись по треугольнику с вершиной в узле, где приложена сила. Кроме того, дополнительно переместятся узлы вследствие упругих деформаций нити.

Если в первом случае величины прогибов зависят только от упругого удлинения нити, то во втором — они связаны с изменением формы и удлинением нити. Очевидно, значения перемещений во втором случае будут намного больше. Если пренебречь перемещениями от упругого удлинения нити, то во втором случае будет иметь место нелинейность второго рода.

Таким образом, статическая работа вантовых покрытий характеризуется нелинейностью второго рода, поскольку перемещения узлов вантовой сети от упругих деформаций нитей при

обычных возможных нагрузках такие же, как и в обычных строительных конструкциях, рассматриваемых как линейно деформируемые.

Иначе говоря, при расчете вантовых сетей нелинейность системы заключается в ее геометрической изменчивости, так как система характеризуется нелинейностью второго рода. Если к вантовой сети приложена равновесная нагрузка, то статическая работа ничем не отличается от геометрически неизменяемых систем. Вантовую сеть на действие равновесной нагрузки можно рассчитывать, пренебрегая нелинейными членами уравнений (23—26). Точность результатов расчета не будет отличаться от точности расчета линейно-деформируемых систем.

Если нагрузка по своему характеру не является равновесной, то расчет вантовой сети можно свести к расчету некоторой другой вантовой системы на равновесную нагрузку путем некоторых несложных формальных преобразований.

Для этого решается предварительная система вспомогательных уравнений:

$$\nabla [H_0 \Delta(z + w)]_l = q_z.$$

Это равносильно определению вертикальных прогибов w для некоторой вспомогательной вантовой сети под действием нагрузки q_z в предположении, что усилия в нитях до и после нагружения останутся равными— H_0 —величинам предварительного натяжения нитей в действительном исходном вантовом покрытии.

В результате решения определится, что $w = z_1$; $u = 0$; $v = 0$.

При подстановке решения в исходные уравнения получим:

$$\nabla [(H_0 + H_1) \Delta(z + z_1)]_l = q_z + q_{z1};$$

$$\nabla [(H_0 + H_1) \cos \Theta]_l = q_x + q_{x1};$$

$$\nabla [(H_0 + H_1) \sin \Theta]_l = q_y + q_{y1};$$

$$H_1 = A \left[\frac{1}{m} \sum_l |\Delta z \cdot \Delta z_1|^m + \frac{1}{2m} \sum_l |\Delta z_1 \cdot \Delta z_1|^m \right],$$

где q_{z1} , q_{x1} , q_{y1} —некоторые невязки, равные

$$q_{z1} = \nabla [H_1 \Delta(z + z_1)]_l;$$

$$q_{x1} = \nabla [H_1 \cos \Theta]_l;$$

$$q_{y1} = \nabla [H_1 \sin \Theta]_l.$$

По сути это означает, что узлы вантовой сети получили вертикальные перемещения z_1 и находятся в положении $(z + z_1)$, в

нитьях возникли дополнительные усилия H_1 и рассматриваемая система находится в равновесии под действием нагрузки

$$q_z + q_{z1}; \quad q_x + q_{x1}; \quad q_y + q_{y1}.$$

Однако в результате расчета должны быть определены перемещения узлов вантовой сети и усилия в нитях под действием заданных нагрузок

$$q_z; \quad q_x; \quad q_y.$$

Если считать положение вантовой сети $(z + z_1)$ исходным, то задача сведется к определению перемещений и усилий в вантовой сети, находящейся под действием нагрузок—

$$-q_{z1}; \quad -q_{x1}; \quad -q_{y1}.$$

Уравнения, выражающие статическую работу вантовой сети, будут выглядеть следующим образом:

$$\nabla [(H_0 + H_1) \Delta w + H \Delta (z + z_1 + w)]_i = -\nabla [H_1 \Delta (z + z_1)]_i;$$

$$\nabla [(H \cos \Theta)_i] = -\nabla [H_1 \cos \Theta]_i;$$

$$\nabla [H \sin \Theta]_i = -\nabla [H_1 \sin \Theta]_i;$$

$$H = A \left[\frac{1}{m} \sum_i |\Delta (z + z_1) \Delta w|_i^m + \frac{1}{2m} \sum_i |\Delta w \cdot \Delta w|_i^m + \Delta^x u \cos \Theta + \Delta^x v \sin \Theta \right].$$

Легко заметить, что для вновь полученной системы нагрузка $(-q_{z1})$ является равновесной (H_1 — в данном случае коэффициент пропорциональности). Следовательно, нелинейными членами в уравнениях системы при решении можно было бы пренебречь. Но величины прогибов $w = z_1$ были найдены при рассмотрении деформаций вспомогательной вантовой сети, в которой величины усилий в нитях не меняются по мере загрузки ее полной нагрузкой q_z . Степень податливости такой системы может быть достаточно высокой и перемещения $w = z_1$ значительными, а следовательно, и невязки $(q_{z1}; \quad q_{x1}; \quad q_{y1})$ будут намного превышать величины заданной нагрузки $(q_z; \quad q_x; \quad q_y)$.

Таким образом свободная от геометрической нелинейности второго рода задача остается нелинейной вследствие появления нелинейности первого рода, связанной с большими значениями новых нагрузок $(-q_{z1}; \quad -q_{x1}; \quad -q_{y1})$.

Если загрузить вспомогательную вантовую сеть нагрузкой q_z , уменьшенной в p раз $(p \cdot q_z)$, то $w = pz_1$; $u = 0$; $v = 0$.

Величина дополнительного усилия определится

$$H_1 = A \left[\frac{p}{m} \sum_i |\Delta z \cdot \Delta z_1|^m + \frac{p^2}{2m} \sum_i |\Delta z_1 \cdot \Delta z_1|^m \right]. \quad (34)$$

Невязки:

$$q_{z1} = \nabla [\Delta (z + pz_1) [H_1 + H_0 (p - 1)]]_i; \quad (35)$$

$$q_{x1} = \nabla [H_1 \cos \Theta]_i; \quad (36)$$

$$q_{y1} = \nabla [H_1 \sin \Theta]_i. \quad (37)$$

Параметр p может быть найден из условия равенства нулю суммарной работы невязок во всех узлах вантовой системы на соответствующих перемещениях z_1 .

$$\sum q_{z1} \cdot z_1 = 0,$$

где n —количество узлов в вантовой сети.

Уравнение это нелинейное относительно параметра p , но поскольку оно одно, решение его не представляет принципиальной сложности.

Теперь исходным положением вантовой сети можно считать $(z + pz_1)$. Уравнения, описывающие статическую работу вантовой сети, примут вид:

$$\nabla [(H_0 + H_1) \Delta w + H \Delta (z + pz_1 + w)]_i = -q_{z1};$$

$$\nabla [H \cos \Theta]_i = -q_{x1};$$

$$\nabla [H \sin \Theta]_i = -q_{y1};$$

$$H = A \left[\frac{1}{m} \sum_i |\Delta (z + pz_1) \Delta w|^m + \frac{1}{2m} \sum_i |\Delta w \cdot \Delta w|^m + \Delta^x u \cos \Theta + \Delta^x v \sin \Theta \right].$$

Для вновь полученной системы нагрузки остались по-прежнему равновесными. Введение параметра q , определяемого из вариационного уравнения (условия равенства нулю суммы работ всех невязок), по сути приводит к появлению невязок по величине одного порядка с заданной нагрузкой.

В результате задача свелась к расчету новой вантовой сети, находящейся в положении $z + pz_1$, на равновесную нагрузку $(-q_{z1}; -q_{x1}; -q_{y1})$ по величине такого же порядка, как и заданная неравновесная. Пренебрегая нелинейными членами,

вновь полученную систему уже линейных уравнений можно записать в следующем виде:

$$\nabla [(H_0 + H_1) \Delta w + H \Delta (z + \rho z_1)]_i = -q_{zi}; \quad (38)$$

$$\nabla [H \cos \Theta]_i = -q_{xi}; \quad (39)$$

$$\nabla [H \sin \Theta]_i = -q_{yi}; \quad (40)$$

$$H = A \left[\frac{1}{m} \sum_i |\Delta (z + \rho z_1) \Delta w|_i^m + \Delta^x u \cos \Theta + \Delta^x v \sin \Theta \right]. \quad (41)$$

Таким образом, любая неравновесная нагрузка достаточно просто приводится к равновесной, а это позволяет решать задачу в целом как линейную, а всю нелинейность выразить в одном уравнении, где неизвестным является параметр ρ .

Глава VII

ОБОБЩЕННЫЕ ПРОГРАММЫ

Статический расчет конструкций рассматриваемыми в настоящей работе методами состоит из нескольких основных этапов.

Поскольку рассчитывается не сооружение, а его идеализированная схема (расчетная модель), выбор ее является важным этапом, предшествующим остальным частям расчета. В каждом конкретном случае инженеру-проектировщику необходимо определить структуру стержневой сетки в расчетной модели, уточнить для каждого стержня необходимость учета различных силовых факторов (осевые, изгибающие усилия в вертикальной и горизонтальной плоскостях), сформулировать граничные условия.

Следующим этапом расчета является составление системы уравнений, описывающих статическую работу расчетной модели, и их решение. Эта часть работы выполняется по типовым программам и электронной вычислительной машине либо оператором-вычислителем на клавишной счетной машине. Предварительно инженер-проектировщик должен заполнить таблицы (массивы) исходных данных, содержащих информацию о рассчитываемой системе.

Информация о структуре стержневой сетки расчетной модели записывается в массив T_1 , в котором номера строк соответствуют номерам стержней, а номера граф — номерам узлов. На схеме сетки предварительно пронумеровываются узлы. Порядок нумерации может быть произвольным, однако рекомендуется нумеровать узлы слева направо и снизу вверх (при этом больший номер всегда соответствует концу стержня, — меньший — его началу). В ячейки массива T_1 записываются (-1) под номером узла, соответствующего началу стержня, и $(+1)$ — концу стержня. В ячейки массива N записываются в порядке

возрастания номера стержней начала нитей. Если все нити системы состоят из одного стержня, в массив N записываются все номера стержней.

Информация о граничных условиях содержится в массивах $w1, u1, v1, \alpha1, \beta1, \gamma1$. Во все ячейки этих массивов записываются (+1), за исключением ячеек, где записываются 0, номера которых соответствуют узлам, в которых перемещения $w, u, v, \alpha, \beta, \gamma$ равны нулю.

Информация о геометрических и физических характеристиках системы вводится в массивы:

A —жесткости стержней на растяжение;

B —жесткости стержней на изгиб в вертикальной плоскости;

C —жесткости стержней на изгиб в горизонтальной плоскости;

λ —длина стержней, либо проекции длин стержней на горизонтальную плоскость;

Θ —углы наклона стержней к оси X ;

z —аппликаты узлов системы.

Информация о расчетных нагрузках содержится в массивах:

qz —вертикальные составляющие нагрузок на узлы системы;

qx —горизонтальные составляющие нагрузок по направлению оси X ;

qy —горизонтальные составляющие нагрузок по направлению оси Y .

Ниже приводятся типовые программы для расчета пологой стержневой системы методом ступенчатого нагружения и для расчета вантовых покрытий методом приведения нагрузки к равновесной (для метода ступенчатого нагружения должна быть задана величина c —количество ступеней нагружения; для частного случая линейно-деформируемой системы $c=1$). Программы охватывают все этапы расчета: от составления системы уравнений и их решения до определения расчетных усилий.

Программы составлены на международном алгоритмическом языке АЛГОЛ-60. В настоящей работе принят язык публикаций с изменениями в эталонном языке. Латинский алфавит дополнен русскими и греческими буквами. В заголовках циклов опущены указания о шаге изменения переменных в тех случаях, когда шаг равен единице.

В программах для сокращения записи указаны только названия процедур, из которых состоит большая часть программы, сами процедуры приводятся отдельно. Процедуры, связанные с решением уравнений, не описываются, так как этот вопрос представляет самостоятельный интерес.

В некоторых системах структура сетки стержней такова, что отдельные ее элементы повторяются много раз. В этом случае целесообразно составлять системы уравнений только для этих отдельных элементов по обычной программе. Переход к полной системе осуществляется с помощью процедуры «Присвоение 3».

К исходной информации о структуре системы в этом случае относятся также массивы $L_1, L_2 \dots L_n$. В эти массивы заносятся номера узлов в полной системе, соответствующие номерам узлов в рассматриваемых элементах. Количество массивов зависит от количества различных элементов в стержневой системе. При таком подходе резко сокращается объем вычислений.

Процедура Составление уравнений 1

Начало

М1: для $k := 1$ до n цикл
для $i := 1$ до m цикл

$$b1[k] := b1[k] + T1[i, k] \uparrow 2 \times \cos \theta[i] \uparrow 2 \times B[i]/\lambda[i];$$

М2: для $k := 1$ до n цикл
для $i := 1$ до m цикл

$$b2[k] := b2[k] + T1[i, k] \uparrow 2 \times \cos \theta[i] \times \sin \theta[i] \times B[i]/\lambda[i];$$

М3: для $k := 1$ до n цикл
для $i := 1$ до m цикл

$$b3[k] := b3[k] + T1[i, k] \uparrow 2 \times \sin \theta[i] \uparrow 2 \times B[i]/\lambda[i];$$

М4: для $k := 1$ до n цикл
для $l := 1$ до n цикл
для $i := 1$ до m цикл

$$T4[l, k] := T4[l, k] + T1[i, k] \uparrow 2 \times T1[i, l] \times \sin \theta[i] \times B[i]/\lambda[i] \uparrow 2;$$

М5: для $k := 1$ до n цикл
для $l := 1$ до n цикл
для $i := 1$ до m цикл

$$T4[l, k+n] := T4[l, k+n] - T1[i, k] \uparrow 2 \times T1[i, l] \times \\ \times \cos \theta[i] \times B[i]/\lambda[i] \uparrow 2;$$

для $k := 1$ до n цикл

Начало

если $b1[k] \times b3[k] \neq b2[k] \uparrow 2$ то на М9 иначе на М 6;

М6: для $i := 1$ до m цикл

$$b11[k] := b11[k] + T1[i, k] \uparrow 2 \times B[i]/\lambda[i];$$

М7: для $i := 1$ до n цикл

$$T\alpha[i, k] := (T4[i, k]/b11[k]) \times \alpha1[k];$$

М8: для $i := 1$ до n цикл

$$T\beta[i, k] := (T4[i, k+n]/b11[k]) \times \beta1[k]; \text{ на М21;}$$

М9: если $b_2[k] \neq 0$ то на М12;

М10: для $i := 1$ до n цикл

$$T\alpha[i, k] := (T4[i, k]/b_3[k]) \times \alpha_1[k];$$

М11: для $i := 1$ до n цикл

$$T\beta[i, k] := (T4[i, k+n]/b_1[k]) \times \beta_1[k]; \text{ на М21;}$$

М12: $b_4[k] := 1/b_1[k];$

М13: $b_5[k] := 1/b_3[k];$

М14: $b_6[k] := b_2[k]/(b_1[k] \times b_3[k]);$

М15: $b_7[k] := 1 - b_6[k] \times b_2[k];$

М16: $b_8[k] := b_5[k]/b_7[k];$

М17: $b_9[k] := b_4[k]/b_7[k];$

М18: $b_{10}[k] := b_6[k]/b_7[k];$

М19: для $i := 1$ до n цикл

$$T\alpha[i, k] := (T4[i, k] \times b_8[k] + T4[i, k+n] \times b_{10}[k]) \times \alpha_1[k];$$

М20: для $i := 1$ до n цикл

$$T\beta[i, k] := (T4[i, k+n] \times b_9[k] + T4[i, k] \times b_{10}[k]) \times \beta_1[k];$$

конец k

М21: для $i := 1$ до m цикл

для $k := 1$ до n цикл

для $l := 1$ до n цикл

$$Qv[i, k] := Qv[i, k] + 2 \times B[i]/\lambda[i] \uparrow 2 (T\alpha[k, l] \times T1[i, l] \uparrow 2 \times \sin \theta[i] - \\ - T\beta[k, l] \times T1[i, l] \uparrow 2 \times \cos \theta[i] - 2 \times T1[i, k]/\lambda[i]);$$

М22: для $k := 1$ до n цикл

для $l := 1$ до n цикл

для $i := 1$ до m цикл

$$T31[k, l] := T31[k, l] + (Qv[i, k] \times T1[i, l]) \times \omega_1[k] \times \omega_1[l];$$

конец процедуры

Процедура Составление уравнений 2

Начало

М1: для $k := 1$ до n цикл

для $i := 1$ до m цикл

$$c_1[k] := c_1[k] + c[i]/\lambda[i] \times (T1[i, k] \uparrow 2);$$

М2: для $k := 1$ до n цикл

для $l := 1$ до n цикл

для $i := 1$ до m цикл

$$T\gamma[l, k] := T\gamma[l, k] + T1[i, k] \uparrow 2 \times T1[i, l] \times \sin \theta[i] \times C[i]/\lambda[i] \uparrow 2 \times \\ \times c1[k] \times \gamma1[k];$$

М3: для $k := 1$ до n цикл
 для $l := 1$ до n цикл
 для $i := 1$ до m цикл

$$T\gamma[l, k+n] := T\gamma[l, k+n] - T1[i, k] \uparrow 2 \times T1[i, l] \times \cos \theta[i] \times \\ \times C[i]/\lambda[i] \uparrow 2 \times c1[k] \times \gamma1[k];$$

М4: для $i := 1$ до m цикл
 для $k := 1$ до n цикл
 для $l := 1$ до n цикл

$$Qh[i, k] := Qh[i, k] + 2 \times C[i]/\lambda[i] \uparrow 2 \times (T\gamma[k, l] \times T1[i, l] \uparrow 2 - \\ - T1[i, k] \times 2 \times \sin \theta[i]/\lambda[i]);$$

М5: для $i := 1$ до m цикл
 для $k := 1$ до n цикл
 для $l := 1$ до n цикл

$$Qh[i, k+n] := Qh[i, k+n] + 2 \times C[i]/\lambda[i] \uparrow 2 \times (T\gamma[k, l+n] \times \\ \times T1[i, l] \uparrow 2 + T1[i, k] \times 2 \times \cos \theta[i]/\lambda[i]);$$

М6: для $k := 1$ до n цикл
 для $l := 1$ до n цикл
 для $i := 1$ до m цикл

$$\{ T32[k, l] := T32[k, l] - Qh[i, k] \times \sin \theta[i] \times T1[i, l] \times u1[k] \times u1[l]; \\ T32[l, k+n] := T32[l, k+n] - Qh[i, k+n] \times \sin \theta[i] \times T1[i, l] \times \\ \times v1[k] \times v1[l]; \}$$

М7: для $k := 1$ до n цикл
 для $l := 1$ до n цикл
 для $i := 1$ до m цикл

$$\{ T32[l+n, k] := T32[l+n, k] + Qh[i, k] \times \cos \theta[i] \times T1[i, l] \times u1[k] \times \\ \times u1[l];$$

$$T32[l+n, k+n] := T32[l+n, k+n] + Qh[i, k+n] \times \cos \theta[i] \times T1[i, l] \times \\ \times v1[k] \times v1[l]; \}$$

конец процедуры

Процедура Составление уравнений 3

Начало

для $i := 1$ до m цикл

$$\Delta w[i] := \Delta u[i] := \Delta v[i] := 0;$$

для $v := z, w, u, v$ цикл

М1: для $i := 1$ до m цикл

для $k := 1$ до n цикл

$$\Delta(v)[i] := \Delta v[i] + v[k] \times T1[i, k]/\lambda[i];$$

Примечание. Здесь применяется переменный идентификатор — прием, расширяющий возможности алгола;
для $k := 1$ до n цикл

$$z[k] := z[k] + \omega[k];$$

М2: для $k := 1$ до n цикл

Начало

для $j := 1$ до m цикл

Начало

для $i := N[j]$ до $N[j+1]-1$ цикл

Начало

$$TH[N[j], k] := 0;$$

$$TH[N[j], k] := TH[N[j], k] + [\Delta z[i] + \Delta \omega[i]] \times A[i] \times T1[i, k]/\lambda[i];$$

$$TH[N[j], k+n] := 0;$$

$$TH[N[j], k+n] := TH[N[j], k+n] + \cos \theta[i] \times A[i] \times T1[i, k]/\lambda[i];$$

$$TH[N[j], k+2n] := 0;$$

$$TH[N[j], k+2n] := TH[N[j], k+2n] + \sin \theta[i] \times A[i] \times T1[i, k]/\lambda[i];$$

конец i

конец j

конец k

для $k := 1$ до n цикл

для $j := 1$ до m цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$\{ TH[i, k] := TH[N[j], k];$$

$$TH[i, k+n] := TH[N[j], k+n];$$

$$TH[i, k+2n] := TH[N[j], k+2n] \};$$

М3: для $k := 1$ до n цикл

Начало

для $l := 1$ до n цикл

Начало

для $i := 1$ до m цикл

Начало

$$T33[l, k] := 0;$$

$$T33[l, k] := T33[l, k] + TH[i, k] \times \omega 1[k] \times \omega 1[l] \times (\Delta z[i] + \Delta w[i]) \times \\ \times T1[i, l];$$

$$T33[l, k+n] := 0;$$

$$T33[l, k+n] := T33[l, k+n] + TH[i, k+n] \times \omega 1[k] \times \omega 1[l] \times \\ \times (\Delta z[i] + \Delta w[i]) \times T1[i, l];$$

$$T33[l, k+2n] := 0;$$

$$T33[l, k+2n] := T33[l, k+2n] + TH[i, k+2n] \times \omega 1[k] \times \\ \times \omega 1[l] \times (\Delta z[i] + \Delta w[i]) \times T1[i, l];$$

конец i

конец l

конец k

М4: для $k := 1$ до n цикл

Начало

для $l := 1$ до n цикл

Начало

для $i := 1$ до m цикл

Начало

$$T33[l+n, k+n] := 0;$$

$$T33[l+n, k+n] := T33[l+n, k+n] + u1[k] \times u1[l] \times \cos \theta[i] \times \\ \times TH[i, k+n] \times T1[i, l];$$

$$T33[l+n, k+2n] := 0;$$

$$T33[l+n, k+2n] := T33[l+n, k+2n] + u1[k] \times u1[l] \times \cos \theta[i] \times \\ \times TH[i, k+2n] \times T1[i, l];$$

конец i

конец l

конец k

М5: для $k := 1$ до n цикл

для $l := 1$ до n цикл

для $i := 1$ до m цикл

$$\{ T33[l+2n, k+2n] := 0;$$

$$T33[l+2n, k+2n] := T33[l+2n, k+2n] + TH[i, k+2n] \times \\ \times v1[k] \times v1[l] \times \sin \theta[i] \times T1[k, l]; \}$$

конец процедуры

Процедура Составление уравнений 4

Начало

для $j := 1$ до $m1$ цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$H[N[j]] := H[N[j]] + A[i] \times (\Delta z[i] \times \Delta \omega[i] + 0,5 \times \Delta \omega[i] \times \Delta \omega[i] + \cos \theta[i] \times \Delta u[i] + \sin \theta[i] \times \Delta v[i]);$$

для $j := 1$ до $m1$ цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$H[i] := H[N[j]];$$

М1: для $k := 1$ до n цикл

для $l := 1$ до n цикл

для $i := 1$ до m цикл

$$T34[k, l] := T34[k, l] + \omega l[k] \times \omega l[l] \times H[i] \times T1[i, k] \times T1[l, l] / \lambda[i];$$

конец процедуры

Процедура Определение усилий

для $v := z, \omega, u, v$ цикл

для $i := 1$ до m цикл

для $k := 1$ до n цикл

$$\Delta v[i] := \Delta v[i] + v[k] \times T1[i, k] / \lambda[i];$$

Начало

М1: для $j := 1$ до $m1$ цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$H[N[j]] := H[N[j]] + A[i] \times (\Delta z[i] \times \Delta \omega[i] + 0,5 \times \Delta \omega[i] \times \Delta \omega[i] + \cos \theta[i] \times \Delta u[i] + \sin \theta[i] \times \Delta v[i]);$$

М2: для $k := 1$ до n цикл

для $i := 1$ до n цикл

$$a[k] := a[k] + Ta[i, k] \times \omega[k];$$

М3: для $k := 1$ до n цикл

для $i := 1$ до n цикл

$$\beta[k] := \beta[k] + T\beta[i, k] \times \omega[k].$$

Примечание. Ниже вычисляются для каждого стержня изгибающие моменты в двух сечениях в начале и в конце стержня. Идентификаторы массива, содержащего моменты в начальных сечениях стержня, имеют индекс n , в конечных сечениях соответственно k ;

М4: для $i := 1$ до m цикл

для $k := 1$ до n цикл

$$Mun[i] := Mun[i] + a[k] \times 0,5 \times (T1[i, k] \uparrow 2 - T1[i, k]) \times \sin \theta[i] - \beta[k] \times 0,5 \times (T1[i, k] \uparrow 2 \times T1[i, k] \times \cos \theta[i] - T1[i, k] \times \omega[k] / \lambda[i]) \times 2B[i] / \lambda[i];$$

М5: для $i := 1$ до m цикл
 для $k := 1$ до n цикл

$$Mek[i] := Mek[i] + [\alpha[k] \times 0,5 \times (T1[i, k] \uparrow 2 + T1[i, k]) \times \sin \Theta[i] - \\ - \beta[k] \times 0,5 \times (T1[i, k] \uparrow 2 + T1[i, k]) \times \cos \Theta[i] - T1[i, k] \times \\ \times \omega[k] / \lambda[i]] \times 2B[i] / \lambda[i];$$

М6: для $k := 1$ до n цикл
 для $i := 1$ до n цикл

$$\gamma[k] := \gamma[k] + T\gamma[i, k] \times u[k] + T\gamma[i, k+n] \times v[k];$$

М7: для $i := 1$ до m цикл
 для $k := 1$ до n цикл

$$Mhn[i] := MHN[i] + [\gamma[k] \times 0,5 \times (T1[i, k] \uparrow 2 \times T1[i, k]) - T1[i, k] \times \\ \times u[k] \times \sin \Theta[i] / \lambda[i] + T1[i, k] \times v[k] \times \cos \Theta[i] / \lambda[i]] \times 2 \times C[i] / \lambda[i];$$

М8: для $i := 1$ до m цикл
 для $k := 1$ до n цикл

$$Mhk[i] := Mhk[i] + [\gamma[k] \times 0,5 \times (T1[i, k] \uparrow 2 + T1[i, k]) - T1[i, k] \times \\ \times u[k] \times \sin \Theta[i] / \lambda[i] + T1[i, k] \times v[k] \times \cos \Theta[i] / \lambda[i]] \times 2 \times C[i] / \lambda[i];$$

конец процедуры

Процедура Присвоение 1

Начало

для $k := 0$ до $n-1$ цикл

Начало

для $i := 0$ до $n-1$ цикл

Начало

$$A[1+3i, 1+3k] := A[1+3i, 1+3k] + T31[1+i, 1+k] + \\ + T33[1+i, 1+k] + T34[1+i, 1+k];$$

$$A[2+3i, 1+3k] := A[2+3i, 1+3k] + T33[1+n+i, 1+k];$$

$$A[3+3i, 1+3k] := A[3+3i, 1+3k] + T33[1+2n+i, 1+k];$$

$$A[1+3i, 2+3k] := A[1+3i, 2+3k] + T33[1+i, 1+n+k];$$

$$A[2+3i, 2+3k] := A[2+3i, 2+3k] + T33[1+n+i, 1+n+k] + \\ + T32[1+i, 1+k];$$

$$A[3+3i, 2+3k] := A[3+3i, 2+3k] + T33[1+2n+i, 1+n+k] + \\ + T32[1+n+i, 1+k];$$

$$A[1+3i, 3+3k] := A[1+3i, 3+3k] + T33[1+i, 1+2n+k];$$

$$A[2+3i, 3+3k] := A[2+3i, 3+3k] + T33[1+n+i, 1+2n+k] + \\ + T32[1+i, 1+n+k];$$

$$A[3+3i, 3+3k] := A[3+3i, 3+3k] + T33[1+2n+i, 1+2n+k] + \\ + T32[1+n+i, 1+n+k];$$

конец i

конец k

конец процедуры

Процедура присвоение 2

Начало

для $k := 0$ до $n-1$ цикл

Начало

$$\omega[1+k] := 0;$$

$$\omega[1+k] := \text{Результаты решения } [1+3k];$$

$$u[1+k] := 0;$$

$$u[1+k] := \text{Результаты решения } [2+3k];$$

$$v[1+k] := 0;$$

$$v[1+k] := \text{Результаты решения } [3+3k];$$

конец k

конец процедуры

Процедура Присвоение 3

Начало

$M1$: для $l := 1$ до n цикл

для $i := 1$ до n цикл

для $k := 1$ до n цикл

$$A[L[i, l], L[k, l]] := A[L[i, l], L[k, l]] + B[i, k];$$

конец процедуры

Программа Ступенчатое загрузеение

Начало

целые i, j, k, l, μ, c ; действительные массивы

$$T1[1:m, 1:n]; TH[1:m, 1:3n]; Qv[1:m, 1:n]; T\alpha, T\beta, T31, \\ T34[1:n, 1:n];$$

$$Qh[1:m, 1:2n]; T4, T\gamma[1:n, 1:2n]; T32[1:2n, 1:2n];$$

$$T33[1:3n, 1:3n]; z, \omega, u, v, qz, qx, qy, b1, b2, b3, b4, b5,$$

$$b11, b6, b7, b8, b9, b10, c1, [1:n], A, B, C, \lambda, \Theta, H, \Delta z, \Delta \omega,$$

$$\Delta u, \Delta v[1:m];$$

для $i := 1$ до m цикл

Начало

$qz[i] := qz[i]/c;$

$qx[i] := qx[i]/c;$

$qy[i] := qy[i]/c;$

конец i

Составление уравнений 1;

Составление уравнений 2;

для $\mu := 1$ до c цикл

Начало

Составление уравнений 3;

Составление уравнений 4;

Присвоение 1;

Решение уравнений;

Присвоение 2;

конец μ

Определение усилий;

конец программы

Программа Приведение к равновесной нагрузке

Начало

целые k, l, i, j ; действительные массивы

$T1[1:m, 1:n]; T34[1:n, 1:n]; Qh[1:m, 1:2n];$

$T4, T7[1:n, 1:2n]; T32[1:2n, 1:2n]; T33[1:3n, 1:3n];$
 $TH[1:m, 1:3n]; z, w, u, v, qz, qx, qy, c1, q1, q2, q3 [1:n];$

$A, C, \lambda, \theta, H, \Delta z, \Delta w, \Delta u, \Delta v [1:m], a1, a2, a3, a4, p;$

для $\mu := 0$ цикл

Начало

Составление уравнений 4;

Присвоение 1;

Решение уравнений;

Присвоение 2;

конец μ

для $v := z$, ω цикл

M1: для $i := 1$ до m цикл

для $k := 1$ до n цикл

$$\Delta v[i] := \Delta v[i] + v[k] \times T1[i, k] / \lambda[i];$$

M2: для $j := 1$ до m цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$H1[N[j]] := H1[N[j]] + A[i] \times \Delta z[i] \times \Delta \omega[i];$$

для $j := 1$ до m цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$H1[i] := H1[N[j]];$$

M3: для $j := 1$ до m цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$H2[N[j]] := H2[N[j]] + A[i] \times 0,5 \times \Delta \omega[i] \times \Delta \omega[i];$$

для $j := 1$ до m цикл

для $i := N[j]$ до $N[j+1]-1$ цикл

$$H2[i] := H2[N[j]];$$

M4: для $k := 1$ до n цикл

Начало

для $i := 1$ до m цикл

Начало

$$q1[k] := q1[k] + H1[i] \times \Delta z[i] \times T1[i, k];$$

$$q2[k] := q2[k] + H1[i] \times \Delta \omega[i] \times T1[i, k] + H2[i] \times \Delta z[i] \times T1[i, k];$$

$$q3[k] := q3[k] + H2[i] \times \Delta \omega[i] \times T1[i, k];$$

конец i

конец k

M8: для $k := 1$ до n цикл

Начало

$$a1 := a1 + q1[k] \times \omega[k];$$

$$a2 := a2 + q2[k] \times \omega[k];$$

$$a3 := a3 + q3[k] \times \omega[k];$$

$$a4 := a4 + qz[k] \times \omega[k];$$

конец k

Решение уравнения вида

$$p(a_1 + a_4) + p^2 a_2 + p^3 a_3 = a_4$$

для $k := 1$ до n цикл

$$\omega[k] := \omega[k] \times p$$

для $\mu := 1$ цикл

Начало

Составление уравнений 2;

Составление уравнений 3;

Составление уравнений 4;

Присвоение 1;

для $k := 1$ до n цикл

для $i := 1$ до m цикл

$$qz[k] := -qz[k] + (\Delta z[i] + \Delta \omega[i]) \times H[i] \times T1[i, k];$$

Решение уравнений;

Присвоение 2;

конец μ

Определение усилий;

конец программы

Глава VIII

ПРИМЕНЕНИЕ ОБОБЩЕННЫХ ПРОГРАММ

При выполнении расчетов возникают неудобства, связанные с формой записи массивов. Общепринятая табличная запись приводит при большом количестве неизвестных к разрастанию размеров расчетных бланков; таблицы получаются разноразмерными. Вследствие этого массивы трудно обозримы, чтение их неудобно.

В настоящей главе при изложении примеров расчета применяется строчная запись таблиц. Суть этой записи заключается в следующем. Каждая строка либо графа двухмерной таблицы рассматривается как отдельный массив и заключается в квадратные скобки, перед которыми ставится идентификатор, где указывается, что данный массив является строкой либо графой, соответствующего номера. Все значения этого массива записываются слева направо и сверху вниз, по десять значений в каждой строчке. Каждый новый массив начинается с красной строчки. Вся таблица в свою очередь заключается в квадратные скобки, перед которыми ставится идентификатор этой таблицы. Приведем пример двухмерной таблицы в общепринятой и в строчной формах записи. Табличная форма записи:

Таблица А

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

При решении задач строительной механики матрицы (таблицы) нередко содержат много нулевых значений. Клетки, в которых содержатся числа, отличные от нуля, как правило, рас-

Строчная форма записи:

	0	1	2	3	4	5	6	7	8	9
A [Стр. 1		1	2	3	4	5	6	7	8	9
10		11	12	13	14	15	16]			
Стр. 2		1	2	3	4	5	6	7	8	9
10		11	12	13	14	15	16]			
Стр. 3		1	2	3	4	5	6	7	8	9
10		11	12	13	14	15	16]]			

полагаются сравнительно неширокой полосой — лентой. В этом случае с целью экономии места целесообразно записывать таблицы в компактной форме в виде двух таблиц: в первой указываются адреса клеток, содержащих ненулевые значения, во второй — значения чисел в этих клетках.

Процедура Компактная запись

Начало целые $i, k, l, m, n, n1$; действительные массивы $A[1 : m, 1 : n]$, адреса A , значения $A[1 : m, 1 : n1]$;

для $i := 1$ до m цикл

начало

$l := 0$;

$k := 0$;

M1: $l := l + 1$

M2: $k := k + 1$

если $k \leq n$ то на M3 иначе на M4

M3: если $A[i, k] \neq 0$ то адреса $A[i, l] := k$;

значения $A[i, l] := A[i, k]$;

на M1; иначе на M2;

M4:

конец i

конец процедуры

Процедура чтение компактной записи

Начало целые $i, k, l, m, n, n1$; действительные массивы $A[1 : m, 1 : n]$, адреса A , значения $A[1 : m, 1 : n1]$;

для $i := 1$ до m цикл

для $l := 1$ до $n1$ цикл

$A[i, l]$, адреса $A[i, l] :=$ значения $A[i, l]$;

конец процедуры.

Проиллюстрируем программу приведения нагрузки к равномерной расчетам вантовой сети квадратной в плане, пролетом 42×42 м (см. рис. 41) под действием равномерно распределенной нагрузки.

Все угловые точки жесткого несмещающегося контура подняты относительно средних. Остальные точки располагаются на прямых, соединяющих угловые точки со средними. Схема имеет четыре оси симметрии (рис. 70). Узлы 4, 6, 7, 9, 10, 11, 12, находящиеся на линиях симметрии, закреплены от горизонтальных

смещений. Узлы 1, 2, 3, относящиеся к контуру, закреплены от всех смещений (см. массив $\omega 1$).

Из расчетной схемы видно, что сеть образована из пяти нитей I [1—4—7—9—11—12]; II [2—5—8—10—11]; III [3—6—8—9] IV [3—5—7]; V [2—4]. Хотя нити II и III образованы стержнями, не находящимися в одной вертикальной плоскости, легко видеть, что усилия в стержнях [3—5] и [5—8], [8—10] и [10—11] равны соответственно между собой из условия равновесия в горизонтальной плоскости.

В массив T1, содержащий информацию о структуре стержневой системы, записаны данные в порядке возрастания для стержней первой нити, затем второй и т. д. Так, первый стержень первой нити находится между узлами 1 и 4, поэтому началу стержня соответствует ячейка T1 [1, 1], а концу стержня — ячейка T1 [1, 4]. Таким образом, в массиве T1 номера строк определяют номера стержня, а ненулевые значения ячеек определяют его начало и конец. В ячейки массива N записаны в соответствии с нумерацией в T1 номера стержней начала нитей. Так, вторая нить начинается с шестого стержня [2—5] (T1 стр 6), поэтому ячейка N [2] содержит число 6. После того как узлы, стержни и нити пронумерованы на схеме, в массивах T1 и N заполняются массивы z , qz , λ , H , A , $\omega 1$, содержащие информацию о геометрических, физических характеристиках системы, величинах преднапряжения, нагрузках и граничных условиях. Для стержней, находящихся на линиях симметрии, записаны половинные жесткости, усилия и нагрузки ($0,5A$; $0,5H$; $0,5qz$). В массиве A содержатся величины $\frac{A}{m1}$, где $m1$ — число стержней, образующих соответствующую нить. Все величины имеют размерность тонны и метры.

Поскольку горизонтальные смещения концов нитей равны нулю, перемещения u и v не рассматриваются.

Ниже следующие массивы вычисляются по программе. Массивы, содержащие только нулевые значения, не приводятся; массивы, значения которых в результате вычислений не меняются, вновь не записываются.

Аналогичная вантовая сеть квадратная в плане пролетом 15×15 м (см. рис. 70) под действием равномерно распределенной нагрузки была рассчитана по программе приведения нагрузки к равновесной. Одновременно был произведен расчет мето-

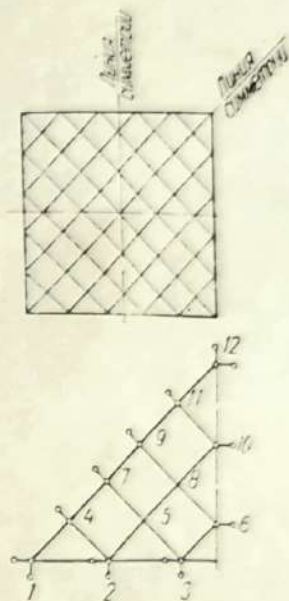


Рис. 70.

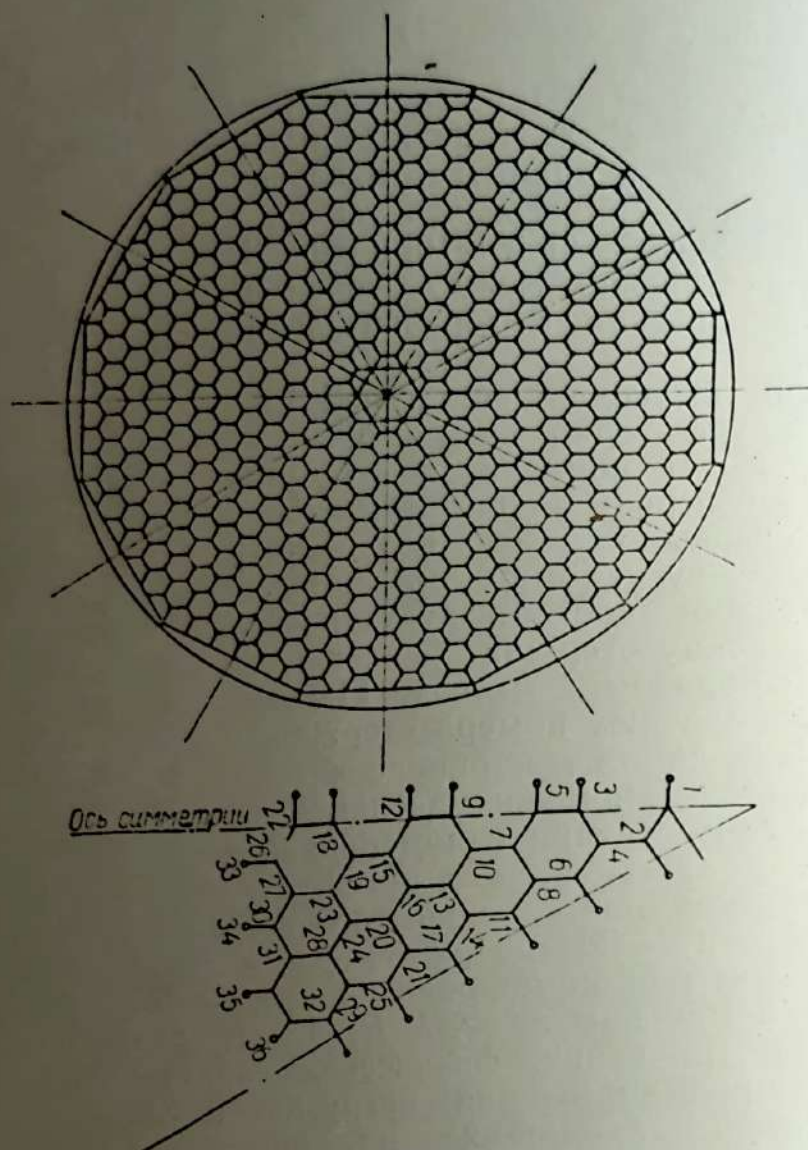


Рис. 71.

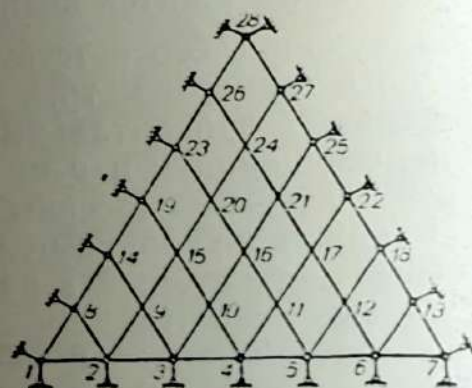
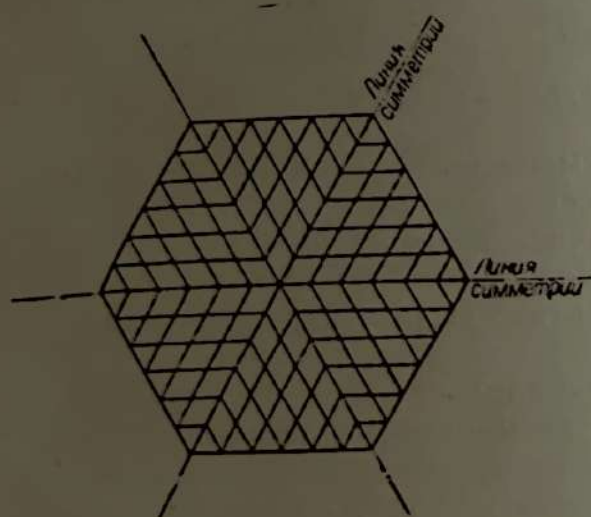


Рис. 72.

дом многоступенчатого нагружения на 10 и на 100 циклов. Расчет методом многоступенчатого нагружения выполнен на машине БЭСМ-2М. Для сравнения приведем результаты расчетов:

Усилия в т	Метод приведения нагрузки к равновесной	7,37	13,24	2,71	5,87	7,65
	Метод многоступенчатой нагрузки 10 циклов	7,31	11,92	4,87	5,26	8,52
	Метод многоступенчатой нагрузки 100 циклов	7,16	11,74	4,61	5,32	8,53

Рассмотрим два случая составления расчетной схемы для сетей.

1. Шестиугольная вантовая сеть (рис. 71) (покрытие крытого рынка в г. Черкассах вариант 2) под действием равномерно распределенной нагрузки. Характерной особенностью такой вантовой сети является постоянная величина распора во всех стержнях сети, поскольку в каждом узле сходятся по три стержня под углом 120° друг к другу. Вследствие симметрии системы рассматривается $1/12$ области сетки. Расчет может быть выполнен по программе приведения нагрузки к равновесной. Ввиду того, что невязки $qz=0$, расчет заканчивается до процедуры составления уравнений 2.

2. Вантовая коосугольная сеть с тросами жесткости пролетом $20,72 \times 24$ м (рис. 72) под действием равномерно распределенной нагрузки. Вследствие симметрии системы рассматривается $1/6$ области. Расчет может

быть выполнен по программе приведения нагрузки к равновесной. Для узлов 8, 14, 19, 23, 26 должны быть приняты граничные условия $u=0,57v$; для узлов 13, 18, 22, 25,

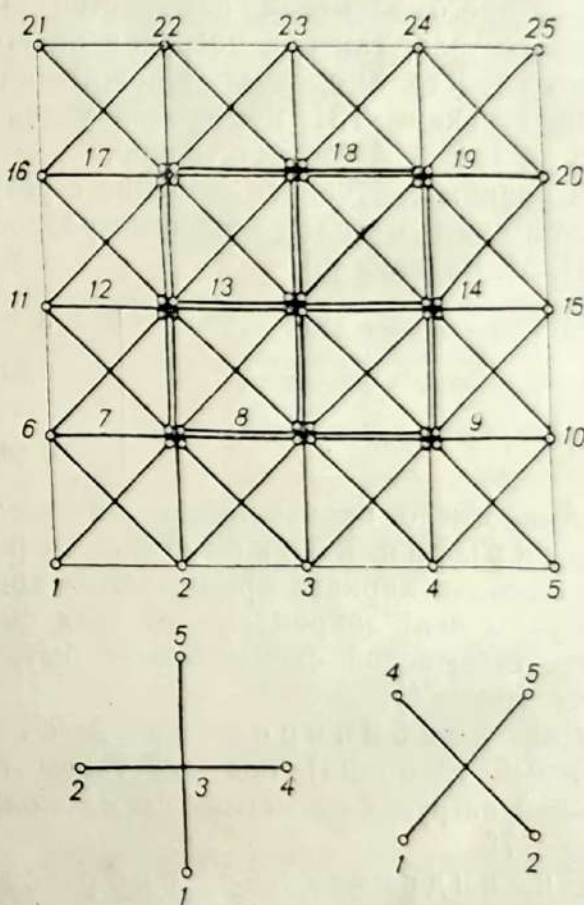


Рис. 73.

27—и = -0,57v. Системы уравнения равновесия преобразовываются: для узлов 8, 14, 19, 23, 26 $\sum i = 0,866[(\sum x) \cdot 0,5 + (\sum y) \cdot 0,866]$ и для узлов 13, 18, 22, 25, 27 $\sum i = +0,866[(\sum x) \cdot 0,5 - (\sum y) \cdot 0,866]$.

Программу ступенчатого нагружения для линейно-деформируемой системы ($c=1$) проиллюстрируем расчетом свободно опертой по контуру плиты. Жесткости на изгиб в горизонтальной плоскости и на осевые силы в этом случае примем равными $C=0$ и $A=0$. Стержневая конечно-разностная модель плиты приведена на рис. 73. Расчет произведем с применением процедуры присвоения 3. Уравнение составим отдельно для одного и другого элементов, из которых образуется стержневая система модели (рис. 73), а затем решение обобщим на всю систему с помощью массивов $L1$ и $L2$. Составление исходных массивов ($T1, \lambda, B, \cos \theta, \sin \theta, \omega 1, \alpha 1, \beta 1$) аналогично составлено в расчете вантовой сети. Учет мембранных усилий может быть произведен по любой из двух программ (в этом случае $c \neq 1$ и $A \neq 0$).

Перед составлением уравнений для системы в целом массив $T31$ второго элемента изменяется исключением перемещения третьего узла, так как нагрузка на этот узел равна нулю. Для этого $T31$ [стр. 3] следует умножить на 0,25 и сложить с остальными строками $T31$, нумерация узлов элемента меняется: 1, 2, 4, 5 на 1, 2, 3, 4 соответственно.

Сравним полученное решение с результатами расчета аналогичной плиты методом конечных разностей, приведенными в книге П. М. Варвака [3].

Прогибы в точках NN	3	4	5
Метод конечных разностей	0,543	0,748	1,034
Расчет плиты как стержневой системы	0,546	0,748	1,029

Аналогично производится расчет ребристых плит перекрытия каркаса на вертикальную нагрузку. Колонны каркаса принимаются абсолютно жесткими. Опорные узлы плит закреплены от всех смещений. В качестве элемента стержневой схемы может быть принята шестигранная плита (рис. 74).

Для комбинированной стержневой системы (лыжный трамплин) под действием горизонтальной и вертикальной нагрузок расчетная схема может быть принята согласно рис. 75.

Цилиндрическая ребристая оболочка под действием равномерно распределенной нагрузки может рассматриваться как стержневая система (рис. 76). Материал ребер и плиты элементов оболочки сконцентрирован в вертикальных и горизонтальных стержнях; арматура плиты элементов, воспринимающая растяжение, сконцентрирована в диагональ-

ных стержнях. Моментная группа усилий должна быть учтена только в одном направлении ввиду наличия шарнирных узлов сопряжений элементов оболочки во втором направлении.

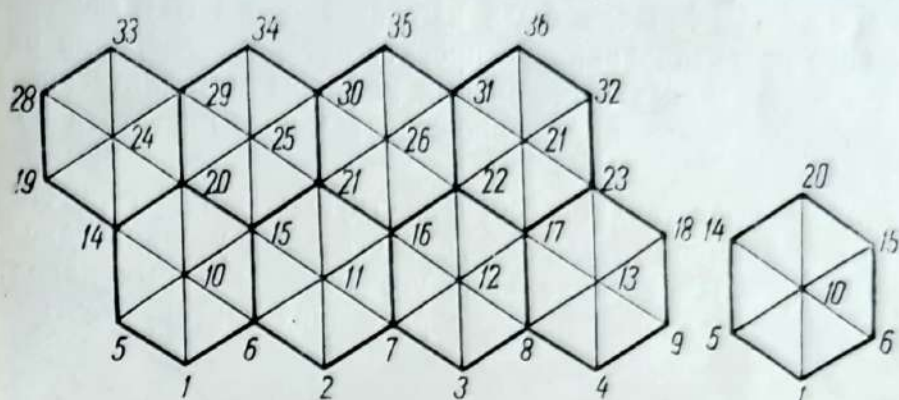


Рис. 74.

Сборная железобетонная ребристая оболочка двоякой кривизны покрытия центрального зала аэровокзала пролетом 48×56 м в г. Борисполе под действием равномерно распределенной нагрузки (рис. 77) рассчитывается

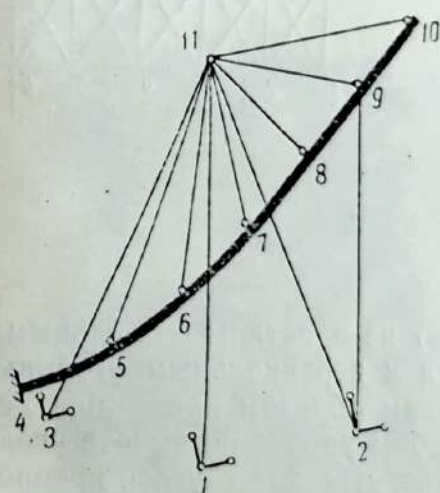


Рис. 75.

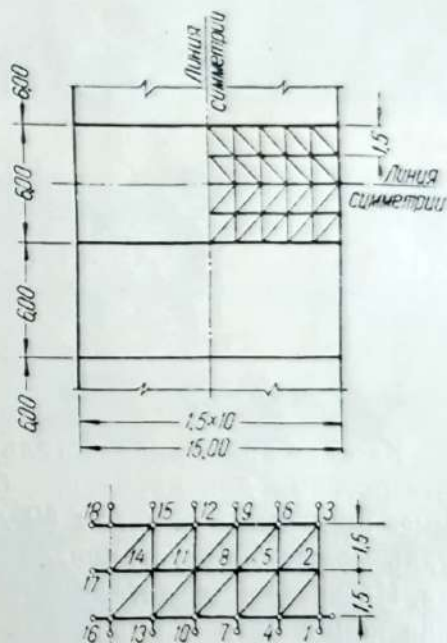


Рис. 76.

в моментной постановке с введением изгибной жесткости ребер в обоих направлениях. Бортовые элементы должны быть приняты упруго-податливыми. Наклон опорных колонн по двум сторонам учитывается граничными условиями: в узлах 1; 7; 13; 19 $w = 0,3249u$. Системы уравнений равновесия преобразовываются $\sum i = 0,9511 (\sum x) + 0,3090 (\sum z)$. Вследствие симметрии системы рассматривается $1/4$ оболочки. В точках 2, 3, 4, 5, 6

имеются вертикальные связи; в точках 1, 7, 13, 19—наклонные связи под углом 18° .

Метод расчета вантовых сетей проверялся на модели. Модель была изготовлена в НИИ экспериментального проектирования по проекту автора. Измерения на модели производились аспирантами кафедры металлических конструкций Киевского инженерно-строительного института Ю. Казанцевым.

Модель представляет собой жесткую двойную горизонтальную раму размерами $2,2 \times 2,2$ м из стальных уголков (рис. 78). К раме крепятся вертикальные уголки, на которых находятся крепежные детали для нитей, образующих вантовую сеть.

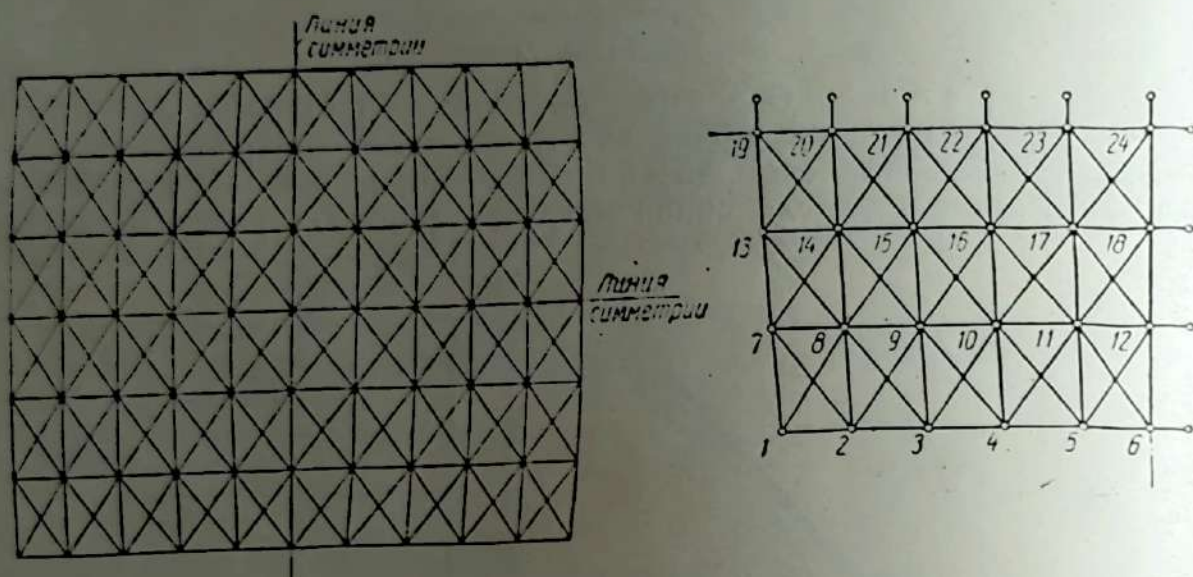


Рис. 77.

Каждая крепежная деталь состоит из пластины с овальными отверстиями для крепления болтами к вертикальным уголкам, имеющим дыры с шагом 50 мм. Таким образом крепежная деталь может быть установлена на любом уровне по высоте уголка. На крепежной детали имеется отверстие для глухого крепления конца нити и ролик для подвижного крепления нити. Ось ролика смазывалась графитовой смазкой.

Вантовая сеть образуется из стальной проволоки $d = 0,0285$ мм, $E = 1,15 \cdot 10^6$ кг/см². Модуль упругости был получен испытанием на растяжение образцов проволоки.

Для получения предварительного натяжения нитей на конец каждой нити, перекинутой через ролик, подвешивались грузы, по весу равные величине предварительного натяжения.

После образования поверхности измерялись аппликаты узлов сети. Затем подвижные концы нитей закреплялись с помо-

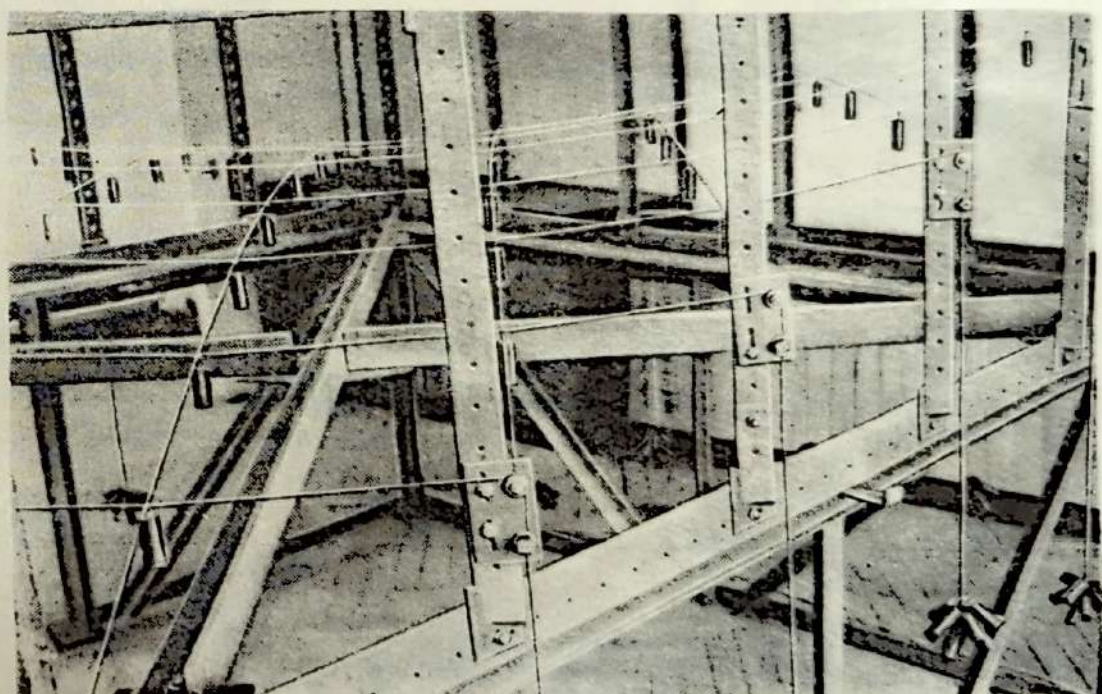
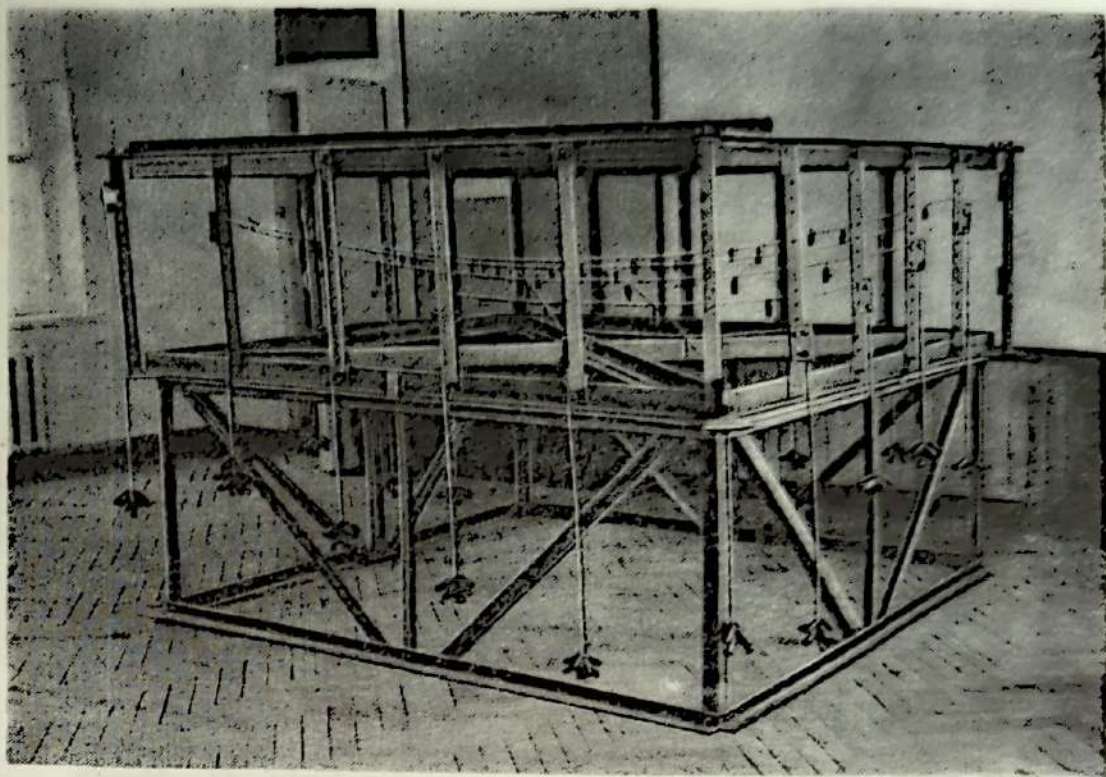


Рис. 78.

щью зажимных болтов и вновь измерялись аппликаты узлов с целью устранения возможных смещений концов нитей при закреплении.

Загрузка производилась грузиками, подвешенными к узлам сети, после чего измерялись аппликаты поверхности. Загрузка и измерение аппликат узлов производились три раза (в дальнейшем необходимость трехкратного загрузки отпала).

Ниже приводятся три примера сравнения экспериментальных данных с результатами расчета (рис. 79).

Пример 1. Вантовая сеть на плоском контуре (рис. 79 а, в).

Пример 2. Вантовая сеть на симметричном контуре (рис. 79 а, б).

Пример 3. Вантовая сеть на кососимметричном контуре (рис. 79 г, д).

Расчет всех примеров производился методом приведения нагрузки к равновесной. Массив Т1 в примерах 1 и 2 такой же, как и в расчете вантовой сети и приведен только в примере 1 в компактной записи.

Расчет валютной сети

0	1	2	3	4	5	6	7	8	9
Т1 [Стр. 1 0,0000	[-1,0000 0,0000	0,0000 0,0000]	0,0000	1,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 2 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	-1,0000	0,0000	0,0000	1,0000	0,0000	0,0000
Стр. 3 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	-1,0000	0,0000	1,0000
Стр. 4 0,0000	[0,0000 1,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-1,0000
Стр. 5 0,0000	[0,0000 -1,0000	0,0000 1,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 6 0,0000	[0,0000 0,0000	-1,0000 0,0000]	0,0000	0,0000	1,0000	0,0000	0,0000	0,0000	0,0000
Стр. 7 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	-1,0000	0,0000	0,0000	1,0000	0,0000
Стр. 8 1,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	-1,0000	0,0000
Стр. 9 -1,0000	[0,0000 1,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 10 0,0000	[0,0000 0,0000	0,0000 0,0000]	-1,0000	0,0000	0,0000	1,0000	0,0000	0,0000	0,0000
Стр. 11 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	-1,0000	0,0000	1,0000	0,0000
Стр. 12 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	-1,0000	1,0000
Стр. 13 0,0000	[0,0000 0,0000	0,0000 0,0000]	-1,0000	0,0000	1,0000	0,0000	0,0000	0,0000	0,0000

Стр. 14	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 15	[0,0000	-1,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	0,0000	0,0000]]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
N Гр.	[1,0000	6,0000	13,000	15,000]	15,000]	15,000]	15,000]	15,000]	15,000]
z Стр.	[-6,0000	-3,6000	-4,0044	-2,4506	-2,4506	-2,4506	-2,4506	-2,4506	-2,4506
-2,2298	-2,2750	-2,2750]	-2,2750]	-2,2750]	-2,2750]	-2,2750]	-2,2750]	-2,2750]	-2,2750]
qz Стр.	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
1,7642	1,7642	0,4410]	1,7642	3,5284	3,5284	3,5284	3,5284	3,5284	3,5284
λ Гр.	[5,9400	5,9400	5,9400	5,9400	5,9400	5,9400	5,9400	5,9400	5,9400
5,9400	5,9400	5,9400	5,9400	5,9400]	5,9400]	5,9400]	5,9400]	5,9400]	5,9400]
H Гр.	[11,880	11,880	11,880	11,880	11,880	11,880	11,880	11,880	11,880
23,760	23,760	23,760	23,760	23,760]	23,760]	23,760]	23,760]	23,760]	23,760]
A Гр.	[1077,1	1077,1	1077,1	1077,1	1077,1	1077,1	1077,1	1077,1	1077,1
1122,1	1122,1	1122,1	1683,2	1683,2]	1683,2]	1683,2]	1683,2]	1683,2]	1683,2]
ω 1 Стр.	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
1,0000	1,0000	1,0000]	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
T 34 [Стр. 4	[0,0000	0,0000	8,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 5	[0,0000	0,0000	0,0000	16,000	16,000	16,000	16,000	16,000	16,000
0,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 6	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 7	[0,0000	0,0000	-2,0000	-4,0000	-4,0000	-4,0000	-4,0000	-4,0000	-4,0000
0,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 8	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
-4,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 9	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
-2,0000	-2,0000	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000

Продолжение расчета вантовой сети

0	1	2	3	4	5	6	7	8	9
Стр. 10 8,0000	[0,0000 -4,0000]	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	-4,0000	0,0000
Стр. 11 -4,0000	[0,0000 8,0000	0,0000 -2,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-2,0000
Стр. 12 0,0000	[0,0000 2,0000	0,0000 2,0000]]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Присвоение I Решение уравнений									
Стр. 2,8414	[0,0000 3,0825	0,0000 3,3030]	0,0000	0,6116	1,1515	1,3002	1,5647	2,1593	2,4620
Δz Гр. -0,0828	[0,3359 -0,0828	0,1997 -0,0304	0,0761 -0,2105	0,0152 -0,0618	0,0000 -0,0681]	0,1935	0,0447	-0,0076	-0,0076
Δw Гр. 0,2188	[0,1029 0,1446	0,1604 0,0509	0,1510 0,1938	0,1044 0,0695	0,0371 0,1029]	0,1938	0,1696	0,1148	0,0405
H 1 Гр. -35,544	[85,893 -35,544	85,893 -35,544	85,893 -75,934	85,893 -75,934	85,893 -23,604]	106,46	106,46	106,46	106,46
H 2 Гр. 40,076	[38,481 40,076	38,481 40,076	38,481 35,700	38,481 35,700	38,481 17,842]	98,397	98,397	98,397	98,397

q1 Стр.	[0,0000	0,0000	13,303	27,126	0,0000	15,313	7,4427	6,3153
—0,0011	0,4979]							
q2 Стр.	[0,0000	0,0000	—3,3411	2,4625	—2,6391	—1,9250	5,5642	3,3143
7,9040	9,9410	3,1892]		6,8187	2,9756	2,8446	9,1494	3,8347
q3 Стр.	[0,0000	0,0000	—0,3737					
7,3059	6,5838	1,4288]						
a 1	[130,55]							
a 2	[78,158]							
a 3	[90,913]							
a 4	[34,066]							

Р е ш е н и е к у б и ч е с к о г о у р а в н е н и я

p	[0,2232]	0,0000	0,1365	0,2570	0,2902	0,3493	0,4820	0,5496
w Стр.	[0,0000	0,0000						
0,6343	0,6881	0,7373]						
Δw Стр.	[0,0229	0,0358	0,0233	0,0082	0,0432	0,0378	0,0256	0,0090
0,0048	0,0322	0,0011	0,0155	0,0229]				
z Стр.	[—6,0000	—3,6000	—3,8679	—2,1936	—1,4021	—2,4685	—1,7025	—1,8158
	—1,5955	—1,5869	—1,5377]					
TH [Стр. 1	[0,0000	0,0000	22,367	0,0000	0,0000	22,800	0,0000	12,931
0,0000	5,4872	1,5033]						
Стр. 2	[0,0000	0,0000	22,367	0,0000	0,0000	22,800	0,0000	12,931
0,0000	5,4872	1,5033]						
Стр. 3	[0,0000	0,0000	22,367	0,0000	0,0000	22,800	0,0000	12,931
0,0000	5,4872	1,5033]						
Стр. 4	[0,0000	0,0000	22,367	0,0000	0,0000	22,800	0,0000	12,931
0,0000	5,4872	1,5033]						

Продолжение расчета вантавой сети

0	1	2	3	4	5	6	7	8	9
Стр. 5 0,0000	[0,0000 5,4872	0,0000 1,5033]	0,0000	22,367	0,0000	0,0000	22,800	0,0000	12,931
Стр. 6 6,7566	[0,0000 0,5916	0,0000 0,0000]	0,0000	0,0000	62,882	0,0000	0,0000	26,378	0,0000
Стр. 7 6,7566	[0,0000 0,5916	0,0000 0,0000]	0,0000	0,0000	62,882	0,0000	0,0000	26,378	0,0000
Стр. 8 6,7566	[0,0000 0,5916	0,0000 0,0000]	0,0000	0,0000	62,882	0,0000	0,0000	26,378	0,0000
Стр. 9 6,7566	[0,0000 0,5916	0,0000 0,0000]	0,0000	0,0000	62,882	0,0000	0,0000	26,378	0,0000
Стр. 10 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	3,0989	0,0000	-5,8911	-3,5702
Стр. 11 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	3,0989	0,0000	-5,8911	-3,5702
Стр. 12 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	3,0989	0,0000	-5,8911	-3,5702
Стр. 13 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	-34,278	0,0000	-13,117	0,0000	0,0000
Стр. 14 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	-34,278	0,0000	-13,117	0,0000	0,0000
Стр. 15 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	-25,559	0,0000	0,0000	0,0000	0,0000	0,0000
Т 33[Стр. 4 0,0000	[0,0000 0,6768	0,0000 0,1854]	0,0000	3,9118	0,0000	0,0000	2,8123	0,0000	1,5950
Стр. 5 1,0413	[0,0000 0,0912	0,0000 0,0000]	0,0000	0,0000	13,838	0,0000	1,5867	4,0653	0,0000

Стр. 6 0,0000	[0,0000 0,0000]	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	0,0513	0,0000	-0,0976	-0,0591
Стр. 7 0,0000	[0,0000 0,6899	0,0000 0,1890]	0,0000	0,0000	2,8123	1,5867	0,0000	3,4738	0,0000	1,6258
Стр. 8 0,4368	[0,0000 0,0382	0,0000 6,0000]	0,0000	0,0000	0,0000	4,0653	-0,0976	0,0000	1,8909	0,1124
Стр. 9 0,0000	[0,0000 0,3913	0,0000 0,1072]	0,0000	0,0000	1,5950	0,0000	-0,0591	1,6258	0,1124	0,9902
Стр. 10 0,1118	[0,0000 0,0098	0,0000 0,0000]	0,0000	0,0000	0,0000	1,0413	0,0000	0,0000	0,4368	0,0000
Стр. 11 0,0098	[0,0000 0,1669	0,0000 0,0424]	0,0000	0,0000	0,6768	0,0911	0,0000	0,6899	0,0382	0,3913
Стр. 12 0,0000	[0,0000 0,0454	0,0000 0,0124]]	0,0000	0,0000	0,1854	0,0000	0,0000	0,1890	0,0000	0,1072
Н Гр. 17,823	[32,968 17,823	32,968 17,823	32,968 8,5901	32,968 8,5901	32,968 19,380]	32,968 19,380]	52,425	52,425	52,425	52,425
Т 34 [Стр. 4 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	14,363	0,0000	0,0000	-5,5502	0,0000	0,0000
Стр. 5 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	20,544	0,0000	-1,4462	-8,8257	0,0000
Стр. 6 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	6,0010	0,0000	-3,0005	0,0000
Стр. 7 0,0000	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	-5,5502	-14,462	0,0000	12,547	0,0000	-5,5502
Стр. 8 -8,8257	[0,0000 0,0000	0,0000 0,0000]	0,0000	0,0000	0,0000	-8,8257	-3,0005	0,0000	23,652	-3,0005

Продолжение расчета вантовой сети

0	1	2	3	4	5	6	7	8	9
Стр. 9 0,0000	[0,0000 -5,5502	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	-5,5502	-3,0005	14,101
Стр. 10 17,651	[0,0000 -8,8257	0,0000 0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000	-8,8257	0,0000
Стр. 11 -8,8257	[0,0000 19,926	0,0000 -5,5502]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-5,5502
Стр. 12 0,0000	[0,0000 -5,5502	0,0000 5,5502]	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000

Присвоение I

qz Стр. -0,8956	[0,0000 -0,6908	0,0000 -0,1678]	0,0000	1,4282	3,5120	-1,4688	1,9832	-0,7006	0,2469
--------------------	---------------------	--------------------	--------	--------	--------	---------	--------	---------	--------

Решение уравнений

х Стр. 0,2381	[0,0000 0,2402	0,0000 0,2753]	0,0000	-0,1167	-0,0909	0,3117	-0,1402	0,1336	0,0665
Δz Гр. -0,0340	[0,3589 -0,0505	0,2355 -0,0190	0,1100 -0,1672	0,0385 -0,0462	0,0082 -0,0451]	0,2367	0,0826	0,0180	0,0014
Δw Гр. 0,0525	[-0,0196 -0,0300	-0,0040 -0,0113	-0,0348 -0,0153	0,0293 -0,0083	0,0059 -0,0196]	-0,0153	0,0378	0,0176	0,0004
Н Гр. 19,864	[31,105 19,864	31,105 19,864	31,105 13,799	31,105 13,799	31,105 23,018]	54,370	54,370	54,370	54,370

Расчет плиты

0	1	2	3	4	5	6	7	8	9
Т 1 [Стр. 1	[-1,0000	0,0000	1,0000	0,0000	0,0000]	Составление уравнений равновесия для первого элемента			
Стр. 2	[0,0000	0,0000	-1,0000	0,0000	1,0000]				
Стр. 3	[0,0000	-1,0000	1,0000	0,0000	0,0000]				
Стр. 4	[0,0000	0,0000	-1,0000	1,0000	0,0000]				
λ Гр.	[1,0000	1,0000	1,0000	1,0000]					
В Гр.	[1,0000	1,0000	1,0000	1,0000]					
cos θ Гр.	[0,0000	0,0000	1,0000	1,0000]					
sin θ Гр.	[1,0000	1,0000	0,0000	0,0000]					
ω 1 Стр.	[1,0000	1,0000	1,0000	1,0000]					
α 1 Стр.	[1,0000	1,0000	1,0000	1,0000]					
β 1 Стр.	[1,0000	1,0000	1,0000	1,0000]					
b 1 Стр.	[0,0000	1,0000	2,0000	1,0000]					
b 2 Стр.	[0,0000	0,0000	0,0000	0,0000]					
b 3 Стр.	[1,0000	0,0000	2,0000	0,0000]					
Т 4 [Стр. 1	[-1,0000	0,0000	-1,0000	0,0000	0,0000]	0,0000	0,0000	0,0000	0,0000
0,0000]									
Стр. 2	[0,0000	0,0000	0,0000	0,0000	0,0000]	0,0000	1,0000	+1,0000	0,0000
0,0000]									
Стр. 3	[+1,0000	0,0000	0,0000	0,0000	-1,0000]	0,0000	-1,0000	0,0000	+1,0000
0,0000]									
Стр. 4	[0,0000	0,0000	0,0000	0,0000	0,0000]	0,0000	0,0000	-1,0000	-1,0000
0,0000]									
Стр. 5	[0,0000	0,0000	0,0000	0,0000	1,0000]	0,0000	0,0000	0,0000	0,0000
0,0000]									
b 11 Стр.	[1,0000	1,0000	0,0000	1,0000	1,0000]				

0	1	2	3	4	5	6	7	8	9
$T \alpha$ [Стр. 1	[-1,0000	0,0000	-0,5000	0,0000	0,0000]				
Стр. 2	[0,0000	0,0000	0,0000	0,0000	0,0000]				
Стр. 3	[+1,0000	0,0000	0,0000	0,0000	-1,0000]				
Стр. 4	[0,0000	0,0000	0,0000	0,0000	0,0000]				
Стр. 5	[0,0000	0,0000	+0,5000	0,0000	1,0000]				
$T \beta$ [Стр. 1	[0,0000	0,0000	0,0000	0,0000	0,0000]				
Стр. 2	[0,0000	1,0000	+0,5000	0,0000	0,0000]				
Стр. 3	[0,0000	-1,0000	0,0000	+1,0000	0,0000]				
Стр. 4	[0,0000	0,0000	-0,5000	-1,0000	0,0000]				
Стр. 5	[0,0000	0,0000	0,0000	0,0000	0,0000]				
$Q \nu$ [Стр. 1	[1,0000	0,0000	-2,0000	0,0000	1,0000]				
Стр. 2	[-1,0000	0,0000	2,0000	0,0000	-1,0000]				
Стр. 3	[0,0000	1,0000	-2,0000	1,0000	0,0000]				
Стр. 4	[0,0000	-1,0000	2,0000	-1,0000	0,0000]				
$T \beta_1$ [Стр. 1	[1,0000	0,0000	-2,0000	0,0000	1,0000]				
Стр. 2	[0,0000	-1,0000	2,0000	-1,0000	0,0000]				
Стр. 3	[2,0000	2,0000	-8,0000	2,0000	2,0000]				
Стр. 4	[0,0000	-1,0000	2,0000	-1,0000	0,0000]				
Стр. 5	[-1,0000	0,0000	2,0000	0,0000	-1,0000]				
$T \gamma$ [Стр. 1	[-1,0000	0,0000	1,0000	0,0000	0,0000]				
Стр. 2	[0,0000	0,0000	-1,0000	0,0000	1,0000]				
Стр. 3	[0,0000	-1,0000	1,0000	0,0000	0,0000]				
Стр. 4	[0,0000	-1,0000	-1,0000	0,0000	0,0000]				
Стр. 5	[0,0000	-1,0000	-1,0000	0,0000	0,0000]				
$\lambda \Gamma p.$	[0,7080	0,7080	0,7080	0,7080]					
$B \Gamma p.$	[1,4160	1,4160	1,4160	1,4160]					
$\sin \theta \Gamma p.$	[0,7080	0,7080	0,7080	0,7080]					
$\cos \theta \Gamma p.$	[0,7080	0,7080	-0,7080	-0,7080]					

Составление уравнений равновесия
для второго элемента

а 1 Стр.	[1,0000	1,0000	1,0000	1,0000]	1,0000]	0,0000	2,0000	0,0000	2,0000	0,0000
а 1 Стр.	[1,0000	1,0000	1,0000	1,0000]	1,0000]	0,0000	0,0000	-2,0000	-2,0000	0,0000
б 1 Стр.	[1,0000	1,0000	1,0000	1,0000]	1,0000]	0,0000	0,0000	2,0000	0,0000	2,0000
б 1 Стр.	[1,0000	1,0000	4,0000	1,0000]	1,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
б 2 Стр.	[1,0000	-1,0000	0,0000	0,0000]	1,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
б 3 Стр.	[1,0000	1,0000	4,0000	1,0000]	1,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
Т 4 [Стр. 1	[2,0000	0,0000	-2,0000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000]										
Стр. 2	[0,0000	-2,0000	-2,0000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000]										
Стр. 3	[2,0000	2,0000	0,0000	0,0000]	-2,0000]	-2,0000	-2,0000	2,0000	0,0000	-2,0000
2,0000]										
Стр. 4	[0,0000	0,0000	2,0000	2,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	2,0000
0,0000]										
Стр. 5	[0,0000	0,0000	2,0000	2,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
-2,0000]										
б 11 Стр.	[2,0000	2,0000	0,0000	2,0000]	2,0000]	2,0000	2,0000	2,0000	2,0000	2,0000
Т а [Стр. 1	[-1,0000	0,0000	-0,5000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 2	[0,0000	-1,0000	-0,5000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 3	[1,0000	1,0000	0,0000	0,0000]	-1,0000]	-1,0000	-1,0000	-1,0000	-1,0000	-1,0000
Стр. 4	[0,0000	0,0000	0,5000	0,5000]	0,5000]	0,5000	0,5000	0,5000	0,5000	0,5000
Стр. 5	[0,0000	0,0000	0,5000	0,5000]	0,5000]	0,5000	0,5000	0,5000	0,5000	0,5000
Т б [Стр. 1	[1,0000	0,0000	0,5000	0,5000]	0,5000]	0,5000	0,5000	0,5000	0,5000	0,5000
Стр. 2	[0,0000	-1,0000	-0,5000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 3	[-1,0000	1,0000	0,0000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 4	[0,0000	0,0000	0,5000	0,5000]	0,5000]	0,5000	0,5000	0,5000	0,5000	0,5000
Стр. 5	[0,0000	0,0000	-0,5000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
Q в [Стр. 1	[4,0000	0,0000	-8,0000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000
Стр. 2	[-4,0000	0,0000	8,0000	0,0000]	0,0000]	0,0000	0,0000	0,0000	0,0000	0,0000

0	1	2	3	4	5	6	7	8	9
Стр. 3	[0,0000	4,0000	-8,0000	4,0000	0,0000]				
Стр. 4	[0,0000	-4,0000	8,0000	-4,0000	0,0000]]				
ТЗ1 [Стр. 1	[-4,0000	0,0000	8,0000	0,0000	-4,0000]				
Стр. 2	[0,0000	-4,0000	8,0000	-4,0000	0,0000]				
Стр. 3	[8,0000	8,0000	-32,0000	8,0000	8,0000]				
Стр. 4	[0,0000	-4,0000	8,0000	-4,0000	0,0000]				
Стр. 5	[-4,0000	0,0000	8,0000	0,0000	0,0000]				
ТЗ1 [Стр. 1	[-2,0000	2,0000	2,0000	-2,0000]	-4,0000]]				
Стр. 2	[2,0000	-2,0000	-2,0000	2,0000]					
Стр. 3	[2,0000	-2,0000	-2,0000	2,0000]					
Стр. 4	[-2,0000	2,0000	2,0000	-2,0000]]					
qz Стр.	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	1,0000	1,0000	1,0000
0,0000	0,0000	1,0000	1,0000	1,0000	0,0000	0,0000	1,0000	1,0000	1,0000
0,0000	0,0000	0,0000	0,0000	0,0000	0,0000]				
L1 [Стр. 1	[2,0000	3,0000	4,0000	7,0000	8,0000	9,0000	12,000	13,000	14,000]
Стр. 2	[6,0000	7,0000	8,0000	11,0000	12,0000	13,0000	16,000	17,000	18,000]
Стр. 3	[7,0000	8,0000	9,0000	12,000	13,000	14,000	17,000	18,000	19,000]
Стр. 4	[8,0000	9,0000	10,000	13,000	14,000	15,000	18,000	19,000	20,000]
Стр. 5	[12,000	13,000	14,000	17,000	18,000	19,000	22,000	23,000	24,000]]
L2 [Стр. 1	[1,0000	2,0000	3,0000	4,0000	6,0000	7,0000	8,0000	9,0000	11,000
12,000	13,000	14,000	16,000	17,000	18,000	19,000]			
Стр. 2	[2,0000	3,0000	4,0000	5,0000	7,0000	8,0000	9,0000	10,000	12,000
13,000	14,000	15,000	17,000	18,000	19,000	20,000]			
Стр. 3	[6,0000	7,0000	8,0000	9,0000	11,000	12,000	13,000	14,000	16,000

Составление уравнений для системы
в целом

17,000		18,000	19,000	21,000	22,000	23,000	24,000]	14,000	15,000	17,000
Стр. 4	[	7,0000	8,0000	9,0000	10,000	12,000	13,000			
18,000		19,000	20,000	22,000	23,000	24,000	25,000]]			
Т 31 [Стр. 7	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-18,000	8,0000	1,0000
0,0000	[	0,0000	8,0000	-2,0000	0,0000	0,0000	0,0000	-1,0000	0,0000	0,0000]
Стр. 8	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	8,0000	-19,000	8,0000
0,0000	[	0,0000	-2,0000	8,0000	-2,0000	0,0000	0,0000	0,0000	-1,0000	0,0000]
Стр. 9	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-1,0000	8,0000	-18,000
0,0000	[	0,0000	0,0000	-2,0000	8,0000	0,0000	0,0000	0,0000	0,0000	-1,0000]
Стр. 12	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	8,0000	-2,0000	0,0000
0,0000	[	0,0000	-19,000	8,0000	-1,0000	0,0000	0,0000	8,0000	-2,0000	0,0000]
Стр. 13	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-2,0000	8,0000	-2,0000
0,0000	[	0,0000	8,0000	-2,0000	8,0000	0,0000	0,0000	-2,0000	8,0000	-2,0000]
Стр. 14	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-2,0000	8,0000
0,0000	[	0,0000	-1,0000	8,0000	-19,000	0,0000	0,0000	0,0000	-2,0000	8,0000]
Стр. 17	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-1,0000	0,0000	0,0000
0,0000	[	0,0000	8,0000	-2,0000	0,0000	0,0000	0,0000	-18,000	8,0000	-1,0000]
Стр. 18	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-1,0000	0,0000
0,0000	[	0,0000	-2,0000	8,0000	-2,0000	0,0000	0,0000	8,0000	-19,000	8,0000]
Стр. 19	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	-1,0000
0,0000	[	0,0000	0,0000	-2,0000	8,0000	0,0000	0,0000	-1,0000	8,0000	-18,0000]]

Результаты

в Стр.	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,5462	0,7478	0,5462
0,0000	[	0,0000	0,7478	1,0294	0,7478	0,0000	0,0000	0,5462	0,7478	0,5462
0,0000	[	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000]

Проверка экспериментальных данных

Пример 1									
Адреса Т1		0	1	2	Значения Т1		0	1	2
Стр. 1		[Стр. 1	[1,0000	4,0000]	[Стр. 1		[Стр. 1	[—1,0000	1,0000]
Стр. 2		Стр. 2	[4,0000	7,0000]	Стр. 2		Стр. 2	[—1,0000	1,0000]
Стр. 3		Стр. 3	[7,0000	9,0000]	Стр. 3		Стр. 3	[—1,0000	1,0000]
Стр. 4		Стр. 4	[9,0000	11,0000]	Стр. 4		Стр. 4	[—1,0000	1,0000]
Стр. 5		Стр. 5	[11,000	12,000]	Стр. 5		Стр. 5	[—1,0000	1,0000]
Стр. 6		Стр. 6	[2,0000	5,0000]	Стр. 6		Стр. 6	[—1,0000	1,0000]
Стр. 7		Стр. 7	[5,0000	8,0000]	Стр. 7		Стр. 7	[—1,0000	1,0000]
Стр. 8		Стр. 8	[8,0000	10,000]	Стр. 8		Стр. 8	[—1,0000	1,0000]
Стр. 9		Стр. 9	[10,000	11,000]	Стр. 9		Стр. 9	[—1,0000	1,0000]
Стр. 10		Стр. 10	[3,0000	6,0000]	Стр. 10		Стр. 10	[—1,0000	1,0000]
Стр. 11		Стр. 11	[6,0000	8,0000]	Стр. 11		Стр. 11	[—1,0000	1,0000]
Стр. 12		Стр. 12	[8,0000	9,0000]	Стр. 12		Стр. 12	[—1,0000	1,0000]
Стр. 13		Стр. 13	[3,0000	5,0000]	Стр. 13		Стр. 13	[—1,0000	1,0000]
Стр. 14		Стр. 14	[5,0000	7,0000]	Стр. 14		Стр. 14	[—1,0000	1,0000]
Стр. 15		Стр. 15	[2,0000	4,0000]]	Стр. 15		Стр. 15	[—1,0000	1,0000]]

0	1	2	3	4	5	6	7	8	9
N Гр.	[1,0000	6,0000	10,000	13,000	15,000]				
qz Стр.	[0,0000	0,0000	0,0000	0,0500	0,1000	0,0500	0,0500	0,1000	0,0500]
0,0500	0,0500	0,0125]							
z Гр.	[28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289]

Продолжение примера 2

0	1	2	3	4	5	6	7	8	9
<i>H</i> Гр.	[0,5000	0,5000	0,5000	0,5000	0,5000	1,0000	1,0000	1,0000	1,0000
1,0000	1,0000	1,0000	1,0000	1,0000	1,0000]				
<i>A</i> Гр.	[116,20	116,20	116,20	116,20	116,20	290,50	290,50	290,50	290,50
387,30	387,30	387,30	581,00	581,00	1162,0]				
<i>z</i> 1 Стр.	[0,0000	0,0000	0,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
1,0000	1,0000	1,0000]							

Результаты

<i>w</i> теорет.	Стр.	[0,0000	0,0000	—0,0406	0,0402	1,4842	—0,1009	0,6641	0,4100
1,9130		2,1545	2,6022]						
<i>w</i> экспер.	Стр.	[0,0000	0,0000	—0,0500	—0,0438	1,7500	0,0125	0,7880	0,9500
2,0000		2,2000	2,7000]						

Пример 3

	0	1	2	Значения T_1	0	1	2
<i>Адреса T_1</i>	[Стр. 1	[1,0000	7,0000]		[Стр. 1	[—1,0000	1,0000]
	Стр. 2	[7,0000	12,000]		Стр. 2	[—1,0000	1,0000]
	Стр. 3	[12,000	16,000]		Стр. 3	[—1,0000	1,0000]
	Стр. 4	[16,000	19,000]		Стр. 4	[—1,0000	1,0000]
	Стр. 5	[19,000	21,000]		Стр. 5	[—1,0000	1,0000]
	Стр. 6	[2,0000	8,0000]		Стр. 6	[—1,0000	1,0000]
	Стр. 7	[8,0000	13,000]		Стр. 7	[—1,0000	1,0000]
	Стр. 8	[13,000	17,000]		Стр. 8	[—1,0000	1,0000]
	Стр. 9	[17,000	20,000]		Стр. 9	[—1,0000	1,0000]
	Стр. 10	[3,0000	9,0000]		Стр. 10	[—1,0000	1,0000]
	Стр. 11	[9,0000	14,000]		Стр. 11	[—1,0000	1,0000]
	Стр. 12	[14,000	18,000]		Стр. 12	[—1,0000	1,0000]

Продолжение примера 3

	0	1	2	0	1	2
Стр. 13	[4,0000		10,000]	Стр. 13	[-1,0000	1,0000]
Стр. 14	[10,000		15,000]	Стр. 14	[-1,0000	1,0000]
Стр. 15	[5,0000		11,000]	Стр. 15	[-1,0000	1,0000]
Стр. 16	[6,0000		11,000]	Стр. 16	[-1,0000	1,0000]
Стр. 17	[11,000		15,000]	Стр. 17	[-1,0000	1,0000]
Стр. 18	[15,000		18,000]	Стр. 18	[-1,0000	1,0000]
Стр. 19	[18,000		20,000]	Стр. 19	[-1,0000	1,0000]
Стр. 20	[20,000		21,000]	Стр. 20	[-1,0000	1,0000]
Стр. 21	[5,0000		10,000]	Стр. 21	[-1,0000	1,0000]
Стр. 22	[10,000		14,000]	Стр. 22	[-1,0000	1,0000]
Стр. 23	[14,000		17,000]	Стр. 23	[-1,0000	1,0000]
Стр. 24	[17,000		19,000]	Стр. 24	[-1,0000	1,0000]
Стр. 25	[4,0000		9,0000]	Стр. 25	[-1,0000	1,0000]
Стр. 26	[9,0000		13,000]	Стр. 26	[-1,0000	1,0000]
Стр. 27	]13,000		16,000 [	Стр. 27	[-1,0000	1,0000]
Стр. 28	[3,0000		8,0000]	Стр. 28	[-1,0000	1,0000]
Стр. 29	[8,0000		12,000]	Стр. 29	[-1,0000	1,0000]
Стр. 30	[2,0000		7,0000]]	Стр. 30	[-1,0000	1,0000]]

0	1	2	3	4	5	6	7	8	9
N Гр. 30,000]	[1,0000	6,0000	10,0000	13,0000	15,0000	16,0000	21,000	25,000	28,000
Зиспер. Стр. 33,007	[0,0000 41,004	10,000 16,021	20,000 22,018	30,000 28,018	40,000 34,010	0,0000 21,030	9,0112 25,031	17,010 29,022	25,009 24,040
26,032	25,044]								
Зиспер. Стр. [0,0000		10,000	20,000	30,000	40,000	0,0000	8,9666	17,066	25,021

0	1	2	3	4	5	6	7	8	9
32,950	40,933	16,108	22,133	27,982	33,983	21,108	25,037	28,975	21,100
26,067	25,083]								
qz Стр.	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0500	0,1000	0,1000
0,1000	0,0500	0,0500	0,1000	0,1000	0,0500	0,0500	0,1000	0,0500	0,0500
0,0500	0,0250]								
λ Гр.	[28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289
28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289
28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289	28,289
28,289]									
H Гр.	[0,5000	0,5000	0,5000	0,5000	0,5000	1,0000	1,0000	1,0000	1,0000
1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,5000	0,5000	0,5000	0,5000
0,5000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
1,0000]									
A Гр.	[137,08	137,08	137,08	137,08	137,08	342,71	342,71	342,71	342,71
456,95	456,95	456,95	685,42	685,42	1370,8	1370,8	1370,8	1370,8	1370,8
1370,8	342,71	342,71	342,71	342,71	456,95	456,95	456,95	685,42	685,42
1370,8]									
w I Стр.	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	1,0000	1,0000	1,0000
1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
1,0000]									
Результаты									
w теорет. Стр.	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,2200	0,1968	0,1550
0,20411	0,2226	0,2816	0,2564	0,2538	0,2583	0,2654	0,2472	0,2604	0,2160
0,2328	0,1860]								
w эксперим. Стр.	[0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,3500	0,4250	0,4750
0,3000	0,3250	0,6750	0,5125	0,3750	0,5000	0,5500	0,3750	0,3500	0,3500
0,3750	0,4000]								

ОПТИМАЛЬНЫЕ ПРОГРАММЫ РАСЧЕТОВ СООРУЖЕНИЙ

ПРИЕМЫ ОРГАНИЗАЦИИ ПАМЯТИ ЭВМ ПРИ РАСЧЕТЕ СТАТИЧЕСКИ НЕОПРЕДЕЛИМЫХ СИСТЕМ

Задача расчета статически неопределимых стержневых систем является одной из наиболее изученных [64]. За последние годы появился ряд работ, посвященных решению этой задачи на ЭВМ, среди которых следует отметить содержательные работы сотрудников ГИПРОТИС [49, 50, 65, 77].

Хотя данную проблему можно считать принципиально решенной, но на практике при решении больших по объему задач (с большим количеством стержней и неизвестных) могут возникнуть затруднения при размещении числового материала в ограниченном объеме оперативной памяти машины. Эти трудности еще больше возрастают при расчете непрерывных систем, например пластин и оболочек, методом сеток [3].

Поэтому важное значение имеют оптимальные программы, позволяющие наиболее эффективно использовать оперативную память машины.

Рассмотрим некоторые примеры оптимализации, имеющие в известной мере общий характер.

Нам представляется, что для наиболее актуальных, часто встречающихся задач можно иметь несколько вариантов программ. Если задача имеет сравнительно небольшой объем и легко вмещается в оперативную память, можно применять обычные программы с простой логической схемой.

Программы с применением специальных приемов экономии памяти машины, уменьшения объема ввода и т. п. несколько усложняют логическую структуру, а иногда и подготовку задачи к решению. Поэтому оптимальные программы целесообразны в тех случаях, когда решаемую задачу нельзя целиком разместить в памяти.

Таким образом наряду с тенденцией охватить единой программой любые случаи, имеется и другая тенденция—некоторая спе-

циализация программ. Правильное решение состоит, по-видимому, в разумном сочетании указанных тенденций.

Рассмотрим расчет статически неопределимых стержневых систем.

При использовании как метода сил, так и метода деформаций, расчет состоит из трех стадий: составление системы уравнений, решение ее и определение усилий.

Составление уравнений метода сил для пространственной стержневой системы с прямолинейными или слабо искривленными стержнями сводится к определению перемещений по формуле Мора [46]

$$\delta_{ik} = \sum \int \frac{M_i^y M_k^y dl}{EI_y} + \sum \int \frac{M_i^z M_k^z dl}{EI_z} + \sum \int \frac{M_i^x M_k^x dl}{GT_x} + \\ + \sum \int \frac{N_i N_k dl}{EF} + \sum \int \frac{\mu_y Q_i^y Q_k^y dl}{GF} + \sum \int \frac{\mu_z Q_i^z Q_k^z dl}{GF}, \quad (42)$$

где i — индекс усилий от единичных неизвестных;
 k — индекс усилий от единичных неизвестных и от нагрузки;

μ_y и μ_z — коэффициенты, зависящие от формы поперечного сечения;

GT_x — жесткость стержня при кручении.

Обычно при расчете статически неопределимых систем вручную интегрирование производится по разным формулам в зависимости от вида эпюры усилий.

При механизации расчета на клавишных автоматах любая эпюра моментов (в том числе и криволинейная) рассматривается как ломаная и вычисления производятся по известной формуле численного интегрирования (см. стр. 25). Это приводит к упрощению программы. Кроме того, учитывая, что эпюры нормальных и поперечных сил, как правило, прямоугольны, отдельную формулу можно использовать для этого случая, что сокращает вычисления [61]. При расчете на ЭВМ уменьшение количества вычислений не столь важно. Поэтому во всех случаях будем считать линию, ограничивающую эпюру, наклонной (см. рис. 8). Тогда формула для определения перемещений примет вид:

$$\delta_{ik} = \sum \int \frac{S_i S_k dl}{c}, \quad (43)$$

где S_i и S_k — могут быть изгибающими и крутящими моментами, продольными и поперечными силами;

c —характеристика жесткости, которой могут быть величины EI , EF , $\frac{GF}{\mu}$ и GT .

В машину вводятся относительные величины $\alpha = \frac{l}{6c}$.

При таком подходе получаем следующую эскизную программу для определения перемещений (см. стр. 26):

$$\Sigma [(2|gr i \times gr k| + |gr i \times gr k|_{\text{диаг}}) \times gr \alpha] S \rightarrow i.kA, \quad (44)$$

где S —исходная матрица усилий.

В результате вычислений по формуле (44) при различных i и k получаем симметричную систему линейных уравнений и свободные члены.

Ввиду того, что графы 1- n матрицы усилий S обычно мало заполнены, целесообразно вводить отдельно массив шкал, указывающих расположение значащих чисел в строках (1—значащее число, 0—нулевой элемент), и отдельно значащие числа. Для каждой строки матрицы отводится шкала-ячейка (рис. 80), в которой записывается восьмеричное число, соответствующее данному набору единиц и нулей (для очень больших задач шкала может состоять не из одной, а из двух-трех ячеек).

Шкала формирования строки:

в двоичном коде									
1	1	0	0	1	1	0	0	0	1
в восьмеричном коде /для ЗВМ "Куб" /									
3	1		4		3				

Рис. 80.

Для усилий от нагрузки и чисел α в памяти резервируется m и $m/2$ ячеек (m —количество сечений).

T_s						$T_s(q)$						
	1	2	3	...	n		1	2	...	q	Участок	α
1						1					1,2	
2						2					3,4	
3						3					...	
4						4					$m-1, m$	
...						...						
$m-1$						$m-1$						
m						m						

Рис. 81.

Так же, как и в программах решения систем уравнений, здесь предусмотрен последовательный ввод исходных усилий от нагрузки в статически определимой системе. После печати иско-
мых величин усилий в статически неопределимой системе авто-
матически вводятся исходные усилия от следующей комбина-
ции загрузки с вытеснением предыдущих чисел до тех пор,

	1	2	...	n
B1				
B2				

Матрица A

1	2	...	n	n+1

Рис. 82.

пока количество загрузений не станет равно заданному числу q .

Таким образом можно получить решение для неограниченного количества комбинаций нагрузок, для единичных нагрузок — ординаты линий влияния.

Порядок вычислений по программе следующий:

1) оперативная память очищается;

2) последовательно формируются парные строки матрицы S (рис. 81) в массивах $B1$ и $B2$ (рис. 82) и вычисляются коэффициенты по формуле (44) с накоплением в матрице A ;

3) выполняется первый этап прямого хода для матрицы A ;

4) вводятся исходные усилия от первой комбинации загрузки;

5) последовательно формируются парные строки матрицы S в массивах $B1$ и $B2$ и выполняется вычисление свободных членов по формуле (44) с накоплением в графе $n+1$ матрицы A ;

6) выполняется второй этап прямого хода и обратный ход. Полученные величины неизвестных замещают свободные члены. При составлении и решении симметричной системы уравнений учитывается ленточность;

7) последовательно формируются строки матрицы S , выполняется вычисление и печать иско-
мых величин усилий, графа свободных членов матрицы A очищается.

Далее циклически повторяются пункты 4—7 для q комбинаций загрузки.

При расчете сооружений методом сил эпюра усилий слева и справа от границы двух участков интегрирования иногда может иметь скачок, чаще скачка нет (рис. 83) и усилия слева и справа одинаковы. Бывает, что эпюра на протяжении участка прямоугольна. Поэтому многие строки в матрице S являются подобными.

Для того, чтобы исключить перфорацию и ввод повторяющихся строк, применяются нулевые шкалы. Если i -ая строка повторяет предыдущую, то в i -ой шкале записывается нулевой код. Последнее означает, что i -ая строка не формируется, а числа в нее засылаются из предыдущей строки.

В исходной матрице S строки 1 и 2 соответствуют участку 1—2, строки 3 и 4—участку 3—4 и т. д. (рис. 81, 83).

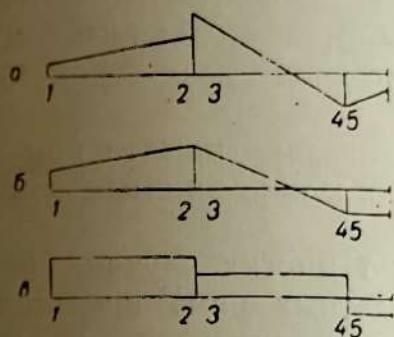


Рис. 83.

	r_i				
	1	2	3	...	n
1					
2, 3, 4					
5, 6					
...					

Рис. 84.

Если нулевых шкал нет, то в строках $B1$ и $B2$ формируются строки 1 и 2, после вычислений по формуле (44)—строки 3 и 4, затем 5 и 6 и т. д.

Если же, например, в шкале 3 записан нуль, то есть строка 3 повторяет строку 2, то вместо операции формирования числа из строки $B2$ засылаются в строку $B1$.

При этом в строке $B1$ будут содержаться числа строки 3, что и требуется.

Если и шкала 4 является нулевой, то есть повторяет строку 3, то числа из строки $B1$ засылаются в строку $B2$. В результате в строке $B2$ будут содержаться числа строки 4.

Таким образом дополнительная проверка равенства нулю содержимого шкалы дает возможность легко учесть всевозможные частные случаи и резко сократить объем вводимой информации.

Облегчается также подготовка задачи к решению, так как уменьшается количество исходных данных. При подготовке матрицы S инженер не записывает дважды в таблицу одни и те же усилия, а лишь указывает номера следующих однотипных строк (рис. 84).

Возможен и другой вариант программы, который отличается от рассмотренного построчным вводом в рабочие строки $B1$ и $B2$ матрицы усилий.

Этот прием был рассмотрен в гл. II на примере первой трансформации Гаусса. Нетрудно заметить, что по формуле (44) осуществляется аналогичное преобразование. По сравнению с вводом матрицы полным массивом построчный ввод ее увеличи-

вает количество обращений к команде ввода, но так как время каждого такого обращения исчисляется микросекундами, то общая разница во времени ничтожна и не имеет практического значения.

Чтобы уменьшить количество печатаемых контрольных сумм, при построчном вводе для них отводится две ячейки — в первой ячейке накапливается контрольная сумма каждой строки (но не печатается), во второй ячейке накапливаются (циклическим сложением) контрольные суммы строк. Содержимое второй ячейки печатается после окончания ввода всех строк матрицы.

Перейдем к описанию приемов организации памяти при составлении программы расчета статически неопределимых стержневых систем методом деформаций.

Как известно, метод деформаций во многих случаях (особенно для многоэтажных и многопролетных рам) предпочтительнее метода сил, так как позволяет резко уменьшить количество неизвестных, что может оказаться существенным даже при выполнении расчетов на ЭВМ.

Кроме того, проще подготавливаются исходные данные — характеристики защемленного стержня.

Использование метода Харди Кросса для расчета рам на ЭВМ (для этого метода разработан ряд программ) нам кажется анахронизмом. Этот метод увеличивает количество вычислений по сравнению с классической формой метода деформаций, так как на каждом цикле определяется вместо одного угла поворота несколько моментов защемления, тогда как эту операцию следует производить только один раз в заключительной стадии [57]. Кроме того, существующие программы, построенные на методе Кросса, ограничиваются рассмотрением рам определенного типа — с вертикальными стойками, одинаковым смещением стоек каждого яруса, отсутствием смещений ригеля и т. п.

Между тем представляет интерес исследование метода деформаций как общего метода расчета рамных систем на электронных вычислительных машинах без указанных ограничений. Единственное ограничение, которое, по-видимому, следует ввести — это допущение о продольной недеформируемости элементов рамы.

В системах, для которых требуется учесть продольные деформации элементов, количество степеней свободы резко возрастает и количество неизвестных в методе деформации становится значительно большим, чем в методе сил (для каждого узла пространственной рамы — шесть степеней свободы, плоской рамы — три степени свободы), но структура уравнений нередко более благоприятна (узкая лента, много нулевых элементов).

При решении систем линейных уравнений метода деформаций можно применять точные и итерационные способы. Как

известно, итерационный способ является весьма эффективным для расчета несвободных и некоторых типов свободных рам (рис. 85). Недостатком является то, что для каждого нагружения расчет нужно фактически выполнять заново, так как инвариантная часть расчета незначительна. Точный метод Гаусса этого недостатка не имеет — расчет может быть выполнен сразу на несколько граф свободных членов, либо сравнительно легко получена обратная матрица.

Следует иметь в виду, что для свободных рам процесс последовательных приближений сходится значительно медленнее, чем для несвободных рам, особенно для систем, имеющих более сложную форму, чем на рис. 85.

При расчете рам на ЭВМ применение итерационного способа имеет то преимущество, что для размещения системы линейных уравнений требуется меньше ячеек памяти, чем при точном способе (если исключить все нулевые элементы [63]).

Однако, если учесть ленточность системы уравнений, то и при способе Гаусса количество ячеек, необходимых для решения системы, резко сокращается.

Структура получаемой системы зависит от принятой нумерации неизвестных. Так, при нумерации, показанной на рис. 85, система уравнений имеет ленточную окаймленную структуру. Порядок системы $n=45$ (36 углов поворота и 9 смещений), порядок ленты $h=5$ и порядок окаймления $n-n_1-h=9$.

При нумерации, показанной на рис. 86, система уравнений имеет ленточную структуру без окаймления (порядок ленты $h=6$), что упрощает логическую схему программы формирования и решения системы уравнений.

Метод Гаусса является наиболее надежным для универсальной программы, основанной на методе деформаций и пригодной для рам различных типов.

При составлении такой программы использовались приемы экономии памяти, во многом сходные с применявшимися при решении систем уравнений и расчете сооружений методом сил.

Положение стержня в системе определяется с помощью оператора—ячейки, в первом адресе которой указывается номер ле-

X_{33}	X_{34}	X_{35}	X_{36}	X_{45}
X_{29}	X_{30}	X_{31}	X_{32}	X_{44}
X_{25}	X_{26}	X_{27}	X_{28}	X_{43}
X_{21}	X_{22}	X_{23}	X_{24}	X_{42}
X_{17}	X_{18}	X_{19}	X_{20}	X_{41}
X_{13}	X_{14}	X_{15}	X_{16}	X_{40}
X_9	X_{10}	X_{11}	X_{12}	X_{39}
X_5	X_6	X_7	X_8	X_{38}
X_1	X_2	X_3	X_4	X_{37}

Рис. 85.

вого конца стержня, в третьем — номер правого конца стержня и во втором — номер перегиба стержня. Например, операторы для стержней 14—19, 15—20 и 18—19 (рис. 86) показаны на рис. 87.

Операторы, количество которых равно числу стержней, записываются при подготовке задачи в восьмеричной системе исчисления и вводятся отдельным массивом.

Для каждого стержня плоской стержневой системы в общем случае имеем следующие характеристики:

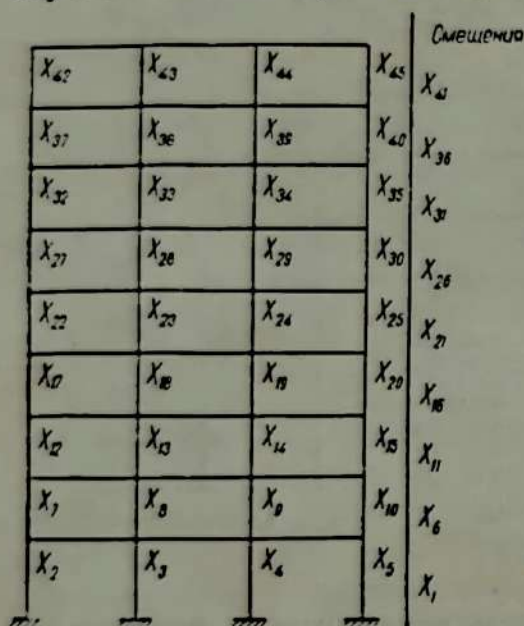


Рис. 86.

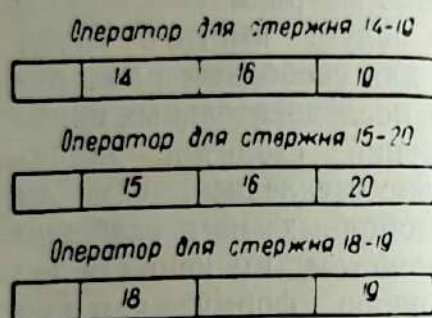


Рис. 87.

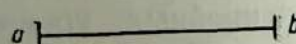


Рис. 88.

1) моменты, возникающие на концах стержня от единичных углов поворота (рис. 88) —

m_{aa} — момент в защемлении a от угла поворота $x_a = 1$;

m_{bb} — момент в защемлении b от угла поворота $x_b = 1$;

$m_{ab} = m_{ba}$ — момент в защемлении a от $x_b = 1$, равный по свойству взаимности моменту в защемлении b от $x_a = 1$;

2) моменты, возникающие от смещения одного конца стержня относительно другого на единицу $m_{a\psi}$, $m_{b\psi}$;

3) перерезывающая сила в сечении узла a от смещения $R_{a\psi}$. По свойству взаимности перерезывающая сила от угла поворота равна моменту от смещения.

Условимся при составлении уравнений равновесия разрезать стержни всегда у левого конца. Благодаря этому отпадает необходимость в указании величин реакций правого конца стержня.

Если стержень не имеет смещения, то во втором адресе оператора ничего не записывается (см. рис. 87), и характеристики, указанные в пунктах 2 и 3, не вводятся в машину.

Вводится признак, по которому выясняется, имеет стержень постоянную жесткость или переменную. Нуль в первом кодовом разряде оператора указывает на постоянную жесткость, единица — на переменную.

Если жесткость постоянная, то $m_{aa} = m_{bb}$;

$$m_{ab} = \frac{m_{aa}}{2}; m_{a\psi} = m_{b\psi}.$$

В этом случае количество вводимых характеристик резко сокращается. Вводятся только величины m_{aa} (погонная жесткость), $m_{a\psi}$, $R_{a\psi}$.

Для остальных величин ячейки памяти не предусматриваются.

Таким образом, будучи вполне общей, программа обладает необходимой гибкостью, позволяющей учесть частные особенности системы и исключить загромождающие память нулевые элементы, а их обычно больше, чем значащих цифр.

Характеристики стержней вводятся отдельным массивом. В соответствии с информацией, указанной в операторах, по программе характеристики из массива выбираются и засылаются в соответствующие ячейки матрицы системы уравнений с накоплением засылаемых величин. Например, коэффициент 8.8 системы уравнений получается последовательным суммированием величин m_{88} для стержней 3—8, 7—8, 8—9 и 8—13. При формировании уравнений порядковый номер данного коэффициента определяется с учетом ленточной структуры матрицы.

Свободные члены (реактивные моменты и реакции от нагрузки) вводятся, как в программах решения систем уравнений (см. гл. III). В ходе расчета машина выдает на печать величины моментов на концах стержней от деформаций (углов поворота и смещений). Для загруженных стержней полученные моменты следует сложить с моментами защемления от нагрузки.

Из изложенного видно, что благодаря развитой логической схеме счета возможности программы значительно расширяются. По существу программа реализует те же операции, которые выполняет инженер: анализирует структуру рамы и упрощает решение в том или ином частном случае в соответствии с самой общей начальной информацией.

Если, например, вторые адреса всех операторов оставить пустыми, то автоматически получим решение для несвободной рамы. Если вторые адреса операторов ригелей оставить пустыми, а во вторых адресах операторов стоек, принадлежащих данному ярусу, записать номер одного и того же неизвестного смещения, то получим решение для многоэтажной, многопролетной рамы с вертикальными стойками (см. рис. 86).

Если элемент рамы получает смещение в нескольких единичных состояниях (что имеет место в рамах с пропущенными ригелями или наклонными стойками), то оператор состоит не из одной, а из двух (редко трех) ячеек. Во второй ячейке (в первом, втором и третьем адресе) записываются номера групповых смещений, относящихся к данному стержню. На наличие следующей ячейки оператора указывает единица в третьем кодовом разряде предыдущей ячейки.

Для группы одинаковых стержней вводятся общие характеристики. На то, что данный стержень повторяет предыдущий, указывает единица во втором кодовом разряде первой ячейки оператора.

К методам строительной механики примыкает близкий им по своей природе метод сеток (метод конечных разностей) [3, 14, 15]. Достоинством этого метода является то, что он легко разрешает математические трудности, связанные с решением сложных краевых задач. Единственная трудность — необходимость составления и решения систем линейных уравнений высоких порядков — в настоящее время успешно преодолевается с помощью быстродействующих электронных вычислительных машин. Метод сеток находит все большее применение при решении сложных задач прикладной теории упругости, в частности, для расчета конструкций типа пластин и оболочек.

В гл. II рассматривались приемы решений систем линейных уравнений высоких порядков на ЭВМ и операторный метод записи матрицы системы уравнений, позволяющий резко сократить трудоемкость подготовки исходных данных.

Как подготавливаются исходные данные, покажем на примере пластинки, работающей на вертикальную нагрузку из плоскости, с густой сеткой (рис. 89).

Требуется подготовить данные для решения системы линейных уравнений 210 порядка. Запись такой системы в обычной форме заняла бы несколько дней и потребовала бы больших площадей бумаги. В операторной форме запись исходных данных занимает немного времени (не более 2—3 часов). В таблице 2 приведены операторы для решения задачи, в данном случае их 25. В кружках записано количество нулей между значащими цифрами.

Например, в развернутой форме оператор 1 имеет вид 18 — 8 1 0 0 0 0 0 0 0 0 0 0 0 0 — 8 2 0 0 0 0 0 0 0 0 0 0 0 0 1.

В табл. 3 помещены номера указателей (210 указателей — по количеству уравнений) и номера соответствующих им операторов. В указанном виде задача подготавливается инженером.

При подготовке задачи для ЭВМ в шкалах операторов вместо значащих цифр ставятся единицы, а значащие цифры помещаются отдельным массивом; указатели и операторы кодиру-

ются в восьмеричной системе исчисления, значащие цифры — в десятичной.

Природа операторов может быть самой разнообразной.

Предлагаемый способ, таким образом, является весьма общим. Однако это не единственный и не всегда лучший путь для частных случаев.

Сетка 15×16

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0—1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0—15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0—20	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
0—43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58
0—57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72
0—71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86
0—85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
0—90	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114
0—113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
0—127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142
0—141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156
0—155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170
0—169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184
0—183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198
0—197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212

Рис. 89.

Если определенный тип задачи настолько важен (и часто применяем), что для него имеет смысл составить отдельную программу, то, анализируя структуру уравнений, можно свести исходную информацию к минимуму.

Например, в работе [16] приведено описание программы для расчета плоской задачи теории упругости — пластинки с отверстием.

Система получаемых уравнений включает всего десять типов значащих чисел (21, —8, 1, 20, 10, 2, —4, 22, 10, 5), а расположение их в матрице следует определенной закономерности, которая отражена в программе. Благодаря этому, исходная информация, характеризующая сетку и достаточная для составления системы уравнений, сводится всего к четырем параметрам.

Таблица 2

№ операции	Операторы строк (граф)														
1	16	-8	1	(11)	-6	2	(12)	1							
2	-8	19	-8	1	(10)	2	-8	2	(12)	1					
3	1	-8	19	-8	1	(10)	2	-8	(12)	1					
4	1	-8	19	-8	(11)	2	-8	2	(12)	1					
5	1	-8	20	(12)	2	-8	(13)	1							
6	-8	2	(12)	19	-8	1	(11)	-8	2	(12)	1				
7	2	-8	2	(11)	-8	20	-8	1	(10)	2	-8	2	(12)	1	
8	2	-8	2	(10)	1	-8	20	-8	1	(10)	2	-8	2	(12)	1
9	2	-8	2	(10)	1	-8	20	-8	(11)	2	-8	2	(12)	1	
10	2	-8	(11)	1	-8	21	(12)	2	-8	(13)	1				
11	1	(13)	-8	2	(12)	19	-8	1	(11)	-8	2	(12)	1		
12	1	(12)	2	-8	2	(11)	-8	20	-8	1	(10)	2	-8	2	(12) 1
13	1	(12)	2	-8	2	(10)	1	-8	20	-8	1	(10)	2	-8	2 (12) 1
14	1	(12)	2	-8	2	(10)	1	-8	20	-8	(11)	2	-8	2	(12) 1
15	1	(12)	2	-8	(11)	1	-8	21	(12)	2	-8	(13)	1		
16	1	(13)	-8	2	(12)	19	-8	1	(11)	-8	2				
17	1	(12)	2	-8	2	(11)	-8	20	-8	1	(10)	2	-8	2	
18	1	(12)	2	-8	2	(10)	1	-8	20	-8	1	(10)	2	-8	2
19	1	(12)	2	-8	2	(10)	1	-8	20	-8	(11)	2	-8	2	
20	1	(12)	2	-8	(11)	1	-8	21	(12)	2	-8				
21	1	(13)	-8	2	(12)	20	-8	1							
22	1	(12)	2	-8	2	(11)	-8	21	-8	1					
23	1	(12)	2	-8	2	(10)	1	-8	21	-8	1				
24	1	(12)	2	-8	2	(10)	1	-8	21	-8					
25	1	(17)	2	-8	(11)	1	-8	22							

Кроме того, нужно ввести лишь величины, характеризующие нагрузку.

Иной характер имеет задача расчета пластинки, работающей из плоскости, со сложными краевыми условиями (край частично закреплен, частично свободен). Здесь количество значащих цифр может быть значительным (появляются дробные чис-

Таблица 3

Указатель строки Оператор	1 1	2 2	3—12 3	13 4	14 5	15 6	16 7	17—26 8	27 9	28 10
Указатель строки Оператор	29 11	30 12	31—40 13	41 14	42 15	43 11	44 12	45—54 13	55 14	56 15
Указатель строки Оператор	57 11	58 12	59—68 13	69 14	70 15	71 11	72 12	73—82 13	83 14	84 15
Указатель строки Оператор	85 11	86 12	87—96 13	97 14	98 15	99 11	100 12	101—110 13	111 14	112 15
Указатель строки Оператор	113 11	114 12	115—124 13	125 14	126 15	127 11	128 12	129—138 13	139 14	140 15
Указатель строки Оператор	141 11	142 12	143—152 13	153 14	154 15	155 11	156 12	157—166 13	167 14	168 15
Указатель строки Оператор	169 11	170 12	171—180 13	181 14	182 15	183 16	184 17	185—194 18	195 19	196 20
Указатель строки Оператор	197 21	198 22	199—208 23	209 24	210 25					

ла в уравнениях для узлов, примыкающих к краю, заранее неизвестно количество, значения и расположение этих чисел). Это затрудняет составление общего алгоритма, основанного на расстановке известных чисел в системе уравнений.

По-видимому, в этом случае операторный метод является целесообразным: в него можно внести упрощения — исключить информацию в виде ячеек указателей и заменить ее соответствующими указаниями в самой программе. Это нетрудно сделать, поскольку, как мы видели, одним оператором описываются десятки уравнений. Тогда исходными данными будет служить небольшое число операторов и значащие цифры к ним.

Изложенные приемы характеризуют в своей совокупности подход к составлению оптимальных программ. При программировании новых задач целесообразно варьировать рассмотренные приемы и пополнять их другими, соответствующими характеру решения.

ОПТИМАЛЬНАЯ ПРОГРАММА РАСЧЕТА ВАНТОВЫХ СИСТЕМ МЕТОДОМ МНОГОСТУПЕНЧАТОГО ЗАГРУЖЕНИЯ

В гл. V—VII был изложен обобщенный метод расчета полных пространственных систем и составлена в общем виде программа, охватывающая разнородные конструкции: ребристые оболочки и плиты, балки-стенки, плоские стержневые системы, рамные каркасы, вантовые системы. Рассмотрен общий случай нелинейной зависимости между усилиями и деформациями, ввиду чего в программу включены приемы линеаризации системы уравнений.

При составлении программы автор или совсем не учитывал, или учитывал лишь частично возможности экономии памяти. Массивы чисел предусматривались для решения задачи в общем виде и имели поэтому весьма большие объемы, которые в современные машины не вмещаются.

Тем не менее такая отвлеченная постановка вопроса является плодотворной по следующим причинам:

1. Дается принципиальная схема программы. В перспективе, ориентируясь на машины недалекого будущего с большим объемом памяти*, эту схему можно будет сохранить, внося в нее такие изменения —

включить ряд признаков, задаваемых в начале расчета, по которым машина сможет распознать, к какому из перечисленных выше классов сооружений относится решаемая задача;

в зависимости от класса решаемой задачи резервируются большие или меньшие массивы памяти и автоматически выполняется ее распределение;

в соответствии с признаком включаются блоки программы, относящиеся к данному классу задач; остальные блоки в решении участия не принимают.

В настоящее время авторами проводится работа по созданию такой универсальной программы.

2. Уже сейчас можно на основе указанной общей программы составить оптимальные программы, позволяющие наиболее рационально использовать память машины, сократить вводную информацию и т. п.

На одной из таких программ, относящейся к расчету вантовых систем в нелинейной постановке методом многоступенчатого нагружения, остановимся ниже. Программа сформулирована на языке АЛГОЛ-60.

Приняты следующие предпосылки: опорный контур предполагается недеформируемым, форма опорного контура и поверхности может быть любой.

*Уже существуют отдельные экземпляры таких машин. Например, американская машина ЛАРК имеет оперативное запоминающее устройство на магнитных сердечниках емкостью 25000 чисел и запоминающее устройство на магнитном барабане емкостью 3 млн. кодов [26].

В основу программы положены формулы (31, 33). С небольшими изменениями эти формулы имеют вид:

$$\sum_k \left[\frac{\Delta^2 w \times H^0}{\lambda} + \frac{1}{\lambda} \frac{EF}{l \lambda} \Delta^2 z \sum_l [\Delta^2 z \times w] \right] = q_k;$$

$$H = H_{\text{пр}} + \sum_l \left[\frac{EF}{l \lambda} w_{\text{пр}} (0,5 \Delta^2 w_{\text{пр}} + \Delta^2 z_{\text{пр}}) \right],$$

где

H^0 — предварительное натяжение;
 $H_{\text{пр}}, z_{\text{пр}}, w_{\text{пр}}$ — величины натяжений в нитях, ординат поверхности, прогибов, найденные на предыдущей ступени расчета;
 $w_{\text{пр}}$ — соответствует величине z_1 в гл. VI;
 $\Delta^2 z (\Delta^2 w)$ — оператор, из которого исключена величина λ , определяется по формуле

$$\Delta^2 z_n = z_{n-1} + 2z_n + z_{n+1};$$

q_k — узловaя нагрузка.

В соответствии с принятым методом расчета основной единицей вантовой системы принята отдельная нить. Расстояния между узлами λ одинаковые в пределах одной нити и могут быть различными в разных нитях.

Положение нити в системе описывается с помощью оператора, представляющего собой группу ячеек: в первой ячейке за-

Оператор нити

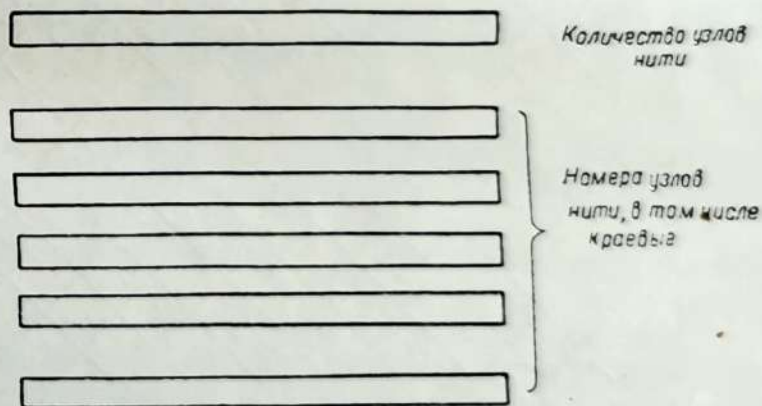


Рис. 90.

писывается количество узлов нити n_2 , в последующих ячейках — номера узлов нити (рис. 90). Легко видеть, что с помощью операторов можно образовать любую форму контура.

Входными данными расчета являются следующие величины:

H^0 — предварительные натяжения нитей;
 z — ординаты поверхности и опорного контура;

- n_1 — число нитей в системе;
 $q = p\lambda_1\lambda_2$ — узловая нагрузка;
 p — нагрузка на 1 м^2 покрытия;
 λ_1, λ_2 — расстояния между узлами пересекающихся нитей;
 n_4 — количество ступеней загрузки;
 n — порядок линеаризованной системы уравнений;
 $\frac{EF}{l_A}$ — приведенная жесткость нити;
 EF — жесткость нити;
 l — длина нити;
 λ — расстояние между узлами нити.

Программа предусматривает либо вычисление начальных ординат поверхности (тогда в соответствующие ячейки вводятся нули), либо ввод известных начальных значений ординат.

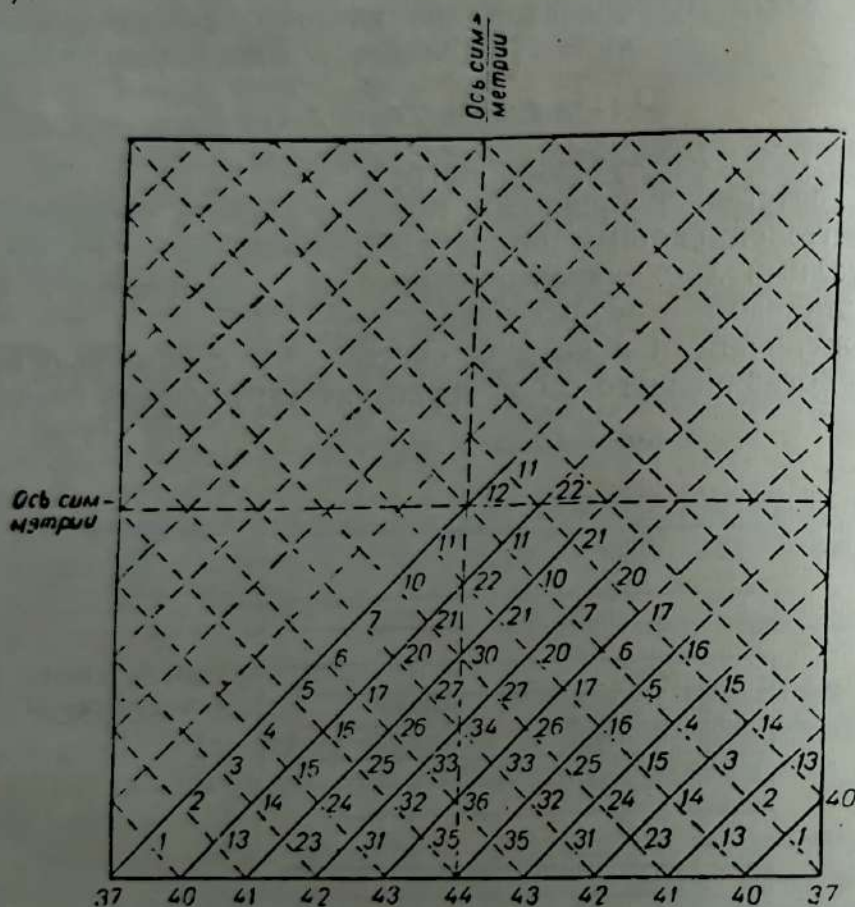


Рис. 91.

Если в симметричной системе нагрузка симметрична (или обратно симметрична), то можно рассматривать только часть поверхности, что значительно уменьшает количество вводимых исходных данных и ускоряет решение. Так, для системы, изображенной на рис. 91, при симметричной нагрузке можно рас-

смагивать $\frac{1}{8}$ часть поверхности. В этом случае оператор нити включает узлы, расположенные до оси симметрии (сплошные линии на рис. 91). Нити второго направления исключаются. Это то же, что рассмагивать нити двух направлений в пределах $\frac{1}{8}$ поверхности.

При вычислении $\frac{EF}{l\lambda}$ берется длина всей нити.

В программу включаются признаки, по которым машина распознает, имеется ли в нити узел, симметричный данному. Единица в первом кодовом разряде узловой ячейки оператора нити указывает на наличие симметричного узла, единица во втором разряде — на то, что ось симметрии расположена между узлами, единица в третьем разряде — на нахождение узла на оси симметрии. Нули в соответствующих разрядах указывают, что симметрии нет или она не используется.

В соответствии с признаками симметрии для некоторых формул вводятся поправочные множители (равные 2 и 1). При этом нарушается взаимность коэффициентов, связывающих узлы, лежащие на оси симметрии, с остальными узлами.

Для восстановления взаимности программой предусмотрено умножение коэффициентов уравнений для узлов, расположенных на оси симметрии, на 0,5. По этой же причине для нити, совпадающей с осью симметрии, жесткости $\frac{EF}{l\lambda}$ и начальные

натяжения вводятся в машину умноженными на 0,5.

Свободные члены уравнений определяются по формуле

$$q_c = \frac{q \times k}{n_4}.$$

Для узлов, лежащих на оси симметрии, $k=0,5$, для центрального узла $k=0,125$ при рассмотрении $\frac{1}{8}$ или 0,250—при $\frac{1}{4}$ поверхности. Для остальных узлов $k=1$.

Свободные члены вводятся в машину в качестве исходных данных.

Решение по программе сводится к многократному (по количеству ступеней загрузки) решению симметричной системы уравнений с меняющимися коэффициентами и постоянной графой свободных членов.

При определении коэффициентов i,k проверяется условие $i \leq k$. Если это условие выполняется, то вычисляется и записывается в матрицу коэффициент i,k , в противном случае—коэффициент k,i .

В памяти предусмотрены ячейки для хранения верхней половины матрицы. В программе решения уравнений треугольная матрица разворачивается в линейную последовательность ячеек.

Так как в основу данной программы положена интегральная зависимость, то матрица получается почти полностью заполненной. Ввиду этого в решении системы линейных уравнений ленточность не используется (параметр h отсутствует).

Выходными результатами расчета являются конечные значения ординат поверхности z_k и усилий в нитях H_k .

Если в начале расчета заданы только ординаты на краях, то печатаются также начальные значения z_n ; упругие прогибы

$$w = z_k - z_n$$

Программа расчета вантовых систем методом многоступенчатого нагружения

Начало Примечание Принято следующее распределение мас-сивов в памяти:

A —матрица системы уравнений,
 $A1$ —упругие прогибы w , свободные члены,
 $A2$ —ординаты поверхности z ,
 $A3$ —начальные натяжения H^0 , приведенные жест-
кости $\frac{EF}{l\lambda}$, расстояния между узлами λ ,
вторые разности $\Delta^2 z$,

Параметры: $A4$ —операторы,
 $n1$ —число нитей,
 n —порядок системы уравнений,
 $n4$ —количество ступеней нагружения.

Переменные: $n2$ —число узлов нити (для разных нитей раз-
личное), $n3$ —количество элементов в строке i системы урав-
нений. Счетчики: m —узлов в нити, $m1$ —
нитей в системе, v —коэффициентов в
уравнении, $m2$ —ступеней нагружения.

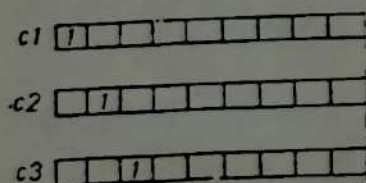


Рис. 92.

Признаки: $\beta=0$ —ввод граничных
условий (в этом случае ординаты по-
верхности определяются решением си-
стемы уравнений Лапласа), $\beta=1$ —
ввод граничных условий и известных
ординат поверхности.

$c1, c2, c3$ —рабочие ячейки (рис. 92), служащие для выделе-
ния содержимых первого, второго и третьего разрядов в узловых
ячейках операторов.

Количество ячеек в матрице (массив A) определяется по
формуле

$$v = 0,5n(n+1).$$

$v1$ —общее количество внутренних узлов во всех нитях.

Операторы кодируются в восьмеричной системе исчисления, остальные данные — в десятичной;

Целые $n, vz, n1, n2, n3, n4, m1, m2, v, \alpha, \beta, c1, c2, c3, v, v0, v1, a1, a2, b, b1, r1, r2, D0, D1, \mu, i, j$;

целый массив $A4[1 : v0]$;

действительные массивы $A[1 : v], A1[1 : 3n], A2[1 : vz], A3[1 : 3n1 + v1]$;

переключатель $S := 31, 38$;

- 1: $m2 := 1$;
- 2: $A[1 : v] := 0$;
- 3: $m1 := 1$;
- 4: $b := 1$;
- 5: $b1 := 3 \times n1$;
- 6: если $\beta = 0$ то на 18;
- 7: $m := 1$;
- 8: $a1 := A3[b1 + m]$;
- 9: $A3[b1 + m] := 2 \times A2[A4[b + m + 1]] - A2[A4[b + m]] - A2[A4[b + m + 2]]$;
- 10: $a2 := A3[b1 + m] - a1$;
- 11: $r1 := A4[b + m + 1] \wedge c1$;
- 12: если $c1 = r1$ то на 14;
- 13: $\alpha := 1$; на 15;
- 14: $\alpha := 2$;
- 15: $A3[m1] := A3[m1] + \alpha \times A3[m1 + n1] \times A1[A4[b + m + 1]] \times (a1 + 0,5 \times a2)$;
- 16: $m := m + 1$;
- 17: если $m \leq A4[b]$ то на 8;
- 18: $m := 1$;
- 19: $v := 0$;
- 20: $r1 := A4[b + m + 1]$;
- 21: $r2 := r1 + v$;
- 22: $\mu := abc(r1 - r2)$;

Примечание μ присваивается абсолютное значение разности $r1$ и $r2$;

- 23: если $r1 \leq r2$ то на 25;
- 24: $i := r2 - 1$; $r3 := r1 \wedge c1$; на 26;
- 25: $i := r1 - 1$; $r3 := r2 \wedge c1$;
- 26: $D0 := 1 + i \times n - i \times (i - 1) / 2$;
- 27: если $\beta = 0$ то на 32; $j := 1$;
- 28: если $c1 = r3$ то на 30;
- 29: $\alpha := 1$; на 39;
- 30: $\alpha := 2$; на 39;
- 31: $A[D0 + \mu] := A[D0 + \mu] + \alpha \times A3[n1 + m1] \times A3[b1 + m] \times A3[b1 + m + v] / A3[2 \times n1 + m1]$;
- 32: если $v = 0$ то на 34;

- 33: если $v=1$ то $\{a:=-1$; на 38} иначе на 45;
 34: $r3:=r1 \wedge c2$; $j:=2$;
 35: если $c2=r3$ то на 37;
 36: $a:=2$; на 39;
 37: $a:=1$; на 39;
 38: $A[D0+\mu]:=A[D0+\mu]+a \times A1[2 \times n+i+1] \times A3[m1]/A3[2 \times n1 + m1]$; на 45;
 39: если $r1 > r1+1$ то на 41;
 40: $r3:=r1 \wedge c3$; на 42;
 41: $r3:=(r1+1) \wedge c3$;
 42: если $c3=r3$ то $a:=a/2$; на $S[j]$;

Примечание 43 и 44 резервные метки;

- 45: $v:=v+1$;
 46: если $m+v \leq A4[b]$ то на 21;
 47: $m:=m+1$;
 48: если $m \leq A4[b]$ то на 19;
 49: $m1:=m1+1$;
 50: $b1:=b1+A4[b]$;
 51: $b:=b+A4[b]+3$;
 52: если $m1 \leq n1$ то на 6;
 53: $A1[1:n]:=A1[n+1:n+n]$;

Примечание На этом заканчивается формирование системы уравнений и начинается ее решение;

Прямой ход:

- 54: $n3:=n1$;
 55: $D0:=1$;
 56: $i:=1$;
 57: $j:=0$;
 58: $D1:=D0$;
 59: $m1:=1$;
 60: $D1:=D1+n3-j$;
 61: $m:=0$;
 62: $r1:=D1+m$;
 63: $A[r1]:=A[r1]-A[D0+m1] \times A[D0+m1+m]/A[D0]$;
 64: $m:=m+1$;
 65: если $m1+m \neq n3-1$ то на 62;
 66: $A1[i+m1]:=A1[i+m1]-A[D0+m1] \times A1[i]/A[D0]$;
 67: $m1:=m1+1$;
 68: если $n3 \leq m1$ то на 70;
 69: $j:=j+1$; на 60;
 70: $D0:=D0+n3$;
 71: $i:=i+1$;
 72: $n3:=n3-1$;
 73: если $n3 \neq 1$ то на 57;

Обратный ход:

- 74: $r1:=n$;

75: $m := 1$;
76: если $m \neq n_3$ то на 80;
77: $A1[r1] := -A1[r1]/A[D0]$;
78: $A4[r1] := A4[r1] + A1[r1]$; на 82;
79: если $D0 = 1$ то на 85;
80: $A1[r1] := A1[r1] + A[D0 + m] \times A1[r1 + m]$;
81: $m := m + 1$; на 76;
82: $n_3 := n_3 + 1$;
83: $D0 := D0 - n_3$;
84: $r1 := r1 - 1$; на 75;
85: если $\beta \neq 0$ то на 90;
86: печать $(A2, 1, n)$;
87: $A1[1 : n] := 0$;
88: чтение $\Pi(A1, n + 1, n + n)$;
89: $\beta := 1$; на 2;
90: $m_2 := m_2 + 1$;
91: если $m_2 \leq n_4$ то на 2;
92: печать $(A2, 1, n)$;
93: печать $(A3, 1, n_1)$;

конец

Глава X

ПЕРЕВОД ПРОГРАММ С ЯЗЫКА АЛГОЛ-60 НА ЯЗЫК ЭВМ

В настоящем разделе рассматриваются лишь те правила перевода, которые необходимы для излагаемых в работе задач.

Применение формальных правил при выполнении перевода программистом создает определенный автоматизм программирования. Естественно, что при этом (так же, как и при машинной автоматизации *) программа занимает несколько больше места, чем при индивидуальном подходе к ее составлению.

При переводе формул с языка АЛГОЛ-60 на язык ЭВМ для переменных величин, обозначенных буквами, резервируются ячейки, адреса которых в начале обозначаются теми же буквами. Идентификаторы массивов заменяются их фиксаторами, то есть адресами ячеек, в которых содержатся начальные адреса массивов. Для получения адреса, в котором содержится адрес текущего элемента массива (адрес второго ранга), следует к фиксатору прибавить выражение, стоящее в индексных (квадратных) скобках.

Составлению рабочей программы предшествует операция распределения адресов — фиксаторам массивов и адресам переменных величин присваиваются их действительные цифровые значения.

Вместо номеров команд рабочей программы, записываются метки формул программы на алголе; в командах передачи управления эти метки заключаются в круглые скобки.

Номера команд записываются после проверки рабочей программы, тогда и метки, фигурирующие в адресах команд, заменяются действительными номерами.

Помимо основной программы, непосредственно переведенной с алгола, рабочая программа содержит еще различные допол-

* Составление полного алгоритма перевода для машины представляет собой сложную задачу, решаемую составителями программирующих программ.

нительные блоки: обращение к компилирующей программе и стандартным подпрограммам (с. п.), в частности к с. п. перевода чисел из десятичной системы в двоичную и обратно и т. п. Именно в этих блоках чаще бывают ошибки. Они встречаются реже после приобретения программистом определенного опыта. Благодаря указанной технологии перевода и тщательной отладке программы на бумаге (см. гл. II) отладка примера на машине занимает немного времени, нередко первая же постановка дает хорошие результаты.

Ниже рассмотрены примеры перевода для машин БЭСМ-2М и «Киев» (системы команд см. [5, 39]).

1. М а ш и н а БЭСМ-2М. Наиболее просто реализуются элементарные операции присваивания типа

$$c := a + b. \quad (45)$$

Заменяя операцию присваивания операцией засылки, получаем адресную формулу $'a + 'b \rightarrow c$. Подставляя вместо a, b, c действительные адреса, например 0021, 0022 и 0023, получим запись в кодах машины:

01	0021	0022	0023
----	------	------	------

Если вместо операции сложения применяются операции вычитания, умножения и деления, то коды операций будут соответственно 02, 03, 04.

Аналогичная формула, но с применением индексных скобок

$$c := T2[i + k] \quad (46)$$

разложится на элементарные адресные формулы:

$$'i + 'k \rightarrow r1; \quad (47)$$

$$'T2 + 'r1 \rightarrow r1; \quad (48)$$

$$'(r1) \rightarrow c. \quad (49)$$

Идентификатор массива заменен его фиксатором, то есть ячейкой, в которой хранится адрес элемента, предшествующего первому элементу массива. Следовательно, $r1$ представляет собой адрес второго ранга.

В кодах машины БЭСМ-2М последние три формулы записываются так:

M1	22	i	k	r1
	22	T2	r1	r1
		0		c

22—операция сложения команд (восьмеричных кодов).

В первом адресе ячейки $M1$ оставлено место (отмеченное овалом), которое должно быть занято величиной $r1$.

Содержимое ячеек $T1, i, k, b$ будем располагать во вторых адресах. Следовательно, для того чтобы получить в первом адресе ячейки $M1$ величину $r1$, нужно сдвинуть содержимое ячейки $r1$ в первый адрес рабочей ячейки $r2$ и последнюю сложить с командой $M1$:

M 1	22	i	k	$r1$
	22	$T2$	$r1$	$r1$
	26	$r1$	0013	$r2$
	22	$M1$	$r2$	$M1$
		○		c
	17	$M1$	$r3$	$M1$

Операция 26 сдвигает содержимое ячейки $r1$ в первый адрес ячейки $r2$, а операция сложения команд 22 производит сложение ячеек $M1$ и $r2$ и засылку результата в ячейку $M1$.

Таким образом вместо овала в ячейке $M1$ теперь находится $r1$, что и требовалось.

Если указанные операции будут повторяться циклически, то после выполнения команды $M1$ ее нужно восстановить в прежнем виде, то есть очистить первый адрес. Это выполняется с помощью операции 17—логического умножения содержимых ячейки $M1$ и константы $r3$.

$r3$	1 . . . 1	0 . . . 0	1 . . . 1	1 . . . 1
------	-----------	-----------	-----------	-----------

Теперь остается присвоить адресам их действительные цифровые значения.

Примем следующее распределение адресов счетчиков и рабочих ячеек:

$T2$ —0020, i —0003, k —0004,
 c —0005, $r1$ —0006, $r2$ —0007,
 $r3$ —0010.

Тогда рассматриваемый отрезок программы примет следующий вид:

M 1

22	0003	0004	0006
22	0020	0006	0006
26	0006	0013	0007
22	M1	0007	M1
	С		0005
17	M1	0010	M1

При распределении адресов следует иметь в виду, что ячейки 0001—0017 применяются в компилирующей программе и содержимое некоторых из них при каждом обращении к стандартным программам (например, перевод из 10→2 и из 2→10) меняется. Поэтому эти ячейки нельзя использовать для хранения величин, которые должны оставаться постоянными при обращениях к стандартным программам (в частности, их нельзя использовать в качестве фиксаторов).

При переводе формулы, записанной на алголе, программист мыслит на адресном языке, так как рабочая программа тоже построена на адресном принципе. Отсюда следует, что будучи конкретным представлением алгола, адресный язык отнюдь не теряет своего значения (по крайней мере до тех пор, пока для всех машин не будут созданы безукоризненно действующие трансляторы, рассчитанные на алгол).

Формулировать обобщенную программу проще и нагляднее на алголе, а читать ее при переводе на язык ЭВМ следует как адресную программу.

Рассмотрим теперь более сложную формулу, записанную на алголе с многократным применением индексных скобок:

$$T1[i + T2[i + k]] : = T1[i + T2[i + k]] + \\ + T3[i + T4[i + T5[k + v]]] / T6[i]. \quad (50)$$

Формула (50) разложена на элементарные адресные выражения в табл. 4.

Однако не нужно думать, что на практике перевод выполняется точно так, как здесь описано. Опытный кодировщик, не затрудняясь, сразу записывает сложную формулу в действительных адресах, минуя запись в символах. Последовательность разложения на элементарные адресные выражения благодаря частому повторению превращается в механическую работу.

Примем следующее распределение адресов:

T1—0020, T2—0021, T3—0022,
T4—0023, T5—0024, T6—0025,

Таблица 4

Формула	Метка	Адресное выражение	Соответствующая часть формулы на алголе
6	M1	'i + 'k → r1	
		'T2 + 'r1 → r1	T2 [i + k]
		'T1 + 'i → r2	
		'r2 + "r1 → r1	T1 [i + T2 [i + k]]
		'k + 'v → r2	
	M2	'T5 + 'r2 → r2	T5 [k + v]
		'T4 + 'i → r3	
		'r3 + "r2 → r2	T4 [i + T5 [k + v]]
	M3	'T3 + 'i → r3	
		'r3 + "r2 → r2	T3 [i + T4 [i + T5 [k + v]]]
	M4	'T6 + 'i → r3	
		"r2/"r3 → r2	T3[i + T4 [i + T5[k + v]]]T6[i]
	M5	'r2 + "r1 → 'r1	Формула (50)

i—0001, k—0002, v—0003,
r1—0004, r2—0005, r3—0006,
r4—0007, r5—0010, r6—0011, r7—0012.

Рабочая программа без нумерации команд (за исключением меток) помещена ниже (табл. 5).

Константы для очистки:

0010	1 . . 1	1 . . 1	0 . . 0	1 . . 1
0011	1 . . 1	0 . . 0	0 . . 0	1 . . 1
0012	1 . . 1	1 . . 1	0 . . 0	0 . . 0

Перевод обобщенной программы выполнен так же, как в предыдущем примере. Для получения последней команды M5 выполняются следующие операции:

- 1) 22—сложить содержимое ячеек M5 и 0004 (второй адрес r1) и результат направить в ячейку M5;
- 2) 26—сдвинуть 0004 из второго в третий адрес ячейки 0007;
- 3) 22—сложить содержимое ячеек M5 и 0007 (третий адрес r1) и результат направить в ячейку M5.

6

6	M1	Для действительных номеров команд	22	0001	0002	0004
			22	0021	0004	0004
			22	0020	0001	0005
			22	M1	0004	M1
			22	0005	○	0004
			22	0002	0003	0005
			22	0024	0005	0005
			22	0023	0001	0006
			22	M2	0005	M2
			22	0006	○	0005
			22	0022	0001	0006
			22	M3	0005	M3
			22	0006	○	0005
			22	0025	0001	0006
			26	0005	0013	0007
			22	M4	0007	M4
			22	M4	0006	M4
			04	○	○	0005
			22	M5	0004	M5
			26	0004	0113	0007
			22	M5	0007	M5
			01	0005	○	○
			17	M1	0010	M1
			17	M2	0010	M2
			17	M3	0010	M3
			17	M4	0011	M4
			17	M5	0012	M5

После выполнения этих операций содержание команды M5 станет следующим:

M5	01	r2	'r1	'r1
----	----	----	-----	-----

Эта команда реализует формулу (50).

Как уже указывалось, на первом этапе команды не нумеруются. Вместо номеров команд слева предусматривается две графы: в первой графе указываются номера формул на алголе (например, для рассматриваемого отрезка программы—формула 6), во второй графе—метки команд, входящих в каждую формулу (в данном случае M1—M5).

После тщательной проверки и отладки программы команды нумеруются и метки формул заменяются номерами команд.

В табл. 5 приведен пример программы, соответствующей очень сложной формуле.

Обычно формуле счета соответствует небольшое количество команд, а логической формуле (логических действий в программах для ЭВМ значительно больше, чем арифметических)—одна, иногда две команды.

Остановимся на некоторых общих операциях, которые не выражаются на алголе (нестандартные для алгола процедуры) и целиком зависят от типа машины:

1. Ввод программы и других данных удобно производить с карты ввода.

В карту ввода (ячейки 0000—0013) записываются команды следующего вида:

30	0100	—	$n - 1$
31		a	c

n —количество вводимых кодов;

a —адрес начала вводимого массива;

c —адрес контрольной суммы.

После ввода карты ввода управление автоматически передается на первую ячейку карты.

Например, для программ, описанных в предыдущих главах, в карте ввода указаны команды ввода исходных параметров программы и самой программы с печатью контрольных сумм (код операции 32).

Ввод данных можно производить с пульта, то есть необходимую информацию набирать непосредственно на пульте.

2. Если в программе имеются обращения к стандартным подпрограммам, то перед началом основной программы записываются четыре команды вызова компилирующей программы.

30	0402	3001	0016
31		0001	0005
77	α_n	α_k	0001
	β_n	β_k	A

α_n, α_k —начальный и конечный адрес основной программы;

β_n, β_k —начальный и конечный адрес вводимых стандартных программ;

$A, A+0157$ —место, отведенное для компилирующей программы.

Обычно компилирующую программу удобно располагать непосредственно за основной программой, а стандартные программы, используемые в основной программе, — в конце памяти.

Компилирующая программа просматривает основную программу, находит в ней команды обращения к стандартным про-

граммам и вызывает их с магнитного барабана в отведенное место памяти машины [32].

В программах, помещенных в следующем разделе, используются только с. п. перевода из $10 \rightarrow 2$ и из $2 \rightarrow 10$, занимающие каждая 57 ячеек (в восьмеричной системе).

Массив памяти, занятый компилирующей программой, необходим только во время ее работы. В дальнейшем на это место можно помещать исходный цифровой материал.

3. Когда это нужно, выполняются обращения к стандартным программам, согласно [32]. Как отмечалось, в наших программах есть обращения только к двум с. п., которые реализуются с помощью команд группового перевода из $10 \rightarrow 2$

77	—	0002	3777
	a_{10}	n	b_2

из $2 \rightarrow 10$

77	—	0004	3777
32	a_2	n	b_{10}

$a_{10}(a_2)$ — адрес начала массива переводимых десятичных (двоичных) чисел;

$b_2(b_{10})$ — адрес начала массива переведенных двоичных (десятичных) чисел;

n — количество ячеек в массиве;

Если не требуется печатать результаты, то код операции печати 32 не пишется.

II. Машина «Киев». В ЭВМ «Киев» (как и в некоторых других современных машинах) имеется специальное запоминающее устройство из одной ячейки — регистр адреса, позволяющее непосредственно оперировать с адресами второго ранга.

Пусть во втором адресе некоторой ячейки A содержится адрес B .

A		B	
-----	--	-----	--

и содержимое этого адреса нужно заслать в ячейку a .

В адресном виде эта операция выглядит так:

$$('A) = 'B \rightarrow a.$$

На машине «Киев» указанные действия реализуются не сложением команд (как в БЭСМ-2М), а с помощью операции 34 —

34	A		a
----	---	--	---

на регистр вызывается содержимое ячейки A —адрес B , и содержимое адреса B засылается в ячейку a . С помощью регистра адреса можно выполнить также операцию формирования команд, избегая при этом сдвигов.

Запись

α	34	4		
$\alpha + 1$	○	4000	○	○

означает, что содержимое адреса A (число B) с регистра засылается в адрес ячейки $\alpha + 1$, в котором стоит признак 4000. Содержимое остальных адресов ячейки $\alpha + 1$ и кода операции может быть любым.

Например, адресная формула " $A + 'b \rightarrow a$ " запишется так:

α	34	A		
$\alpha + 1$	01	4000	b	a

а для адресной формулы " $A + 'b \rightarrow 'A$ " признак 4000 будет в первом и третьем адресе.

α	34	A		
$\alpha + 1$	01	4000	b	4000

Отрезок программы, помещенный в табл. 5, в кодах ЭВМ «Киев» запишется так (табл. 6).

В данном примере благодаря применению операции 34 количество команд для ЭВМ «Киев» существенно меньше, чем для БЭСМ-2М.

Общие операции, не описываемые алголом для ЭВМ «Киев».

1. Двоичные коды (на бланке кодируются восьмеричными числами) вводятся с помощью команды 20, а десятичные—21.

20	a_1	a_2	a_3
21	a_1	a_2	a_3

a_1 —начальный адрес вводимого массива;
 a_2 —конечный адрес вводимого массива;
 a_3 —номер зоны ленты, на которой набит вводимый материал.

Таблица 6

6	Для действительных адресов	01	0001	0002	0004
		01	0021	0004	0004
		01	0020	0001	0005
		34	0004		
		01	0005	4000	0004
		01	0002	0003	0005
		01	0024	0005	0005
		01	0023	0001	0006
		34	0005		
		01	0006	4000	0005
		01	0022	0001	0006
		34	0005		
		01	0006	4000	0005
		01	0025	0001	0006
		34	0005		0005
		34	0006		
		12	0005	4000	0005
		34	0004		
		01	0005	4000	4000

Машина устроена так, что при десятичном вводе управление передается на стандартную программу перевода из 10→2. Стандартные программы в машине «Киев» не занимают места в оперативной памяти, а хранятся в дополнительной так называемой постоянной памяти машины.

2. Вывод чисел на печать может быть в восьмеричной или десятичной системе исчисления, в зависимости от чего меняется положение тумблера печати на печатающем устройстве. В плане счета необходимо указывать ПЧ8 или ПЧ10. Команда печати имеет вид:

22	a_1	a_2	a_3
----	-------	-------	-------

a_1 и a_2 —начальный и конечный адреса печатаемого массива.

a_3 —команда, которой передается управление после печати.

3. Команды обращения к стандартным программам. В наших программах используются только команды 30 обращения и с. п. перевода из 2→10, а именно:

30	3026	a	3100
----	------	-----	------

a —адрес команды, которой передается управление после выполнения стандартной программы;

3100—первая команда стандартной программы.

Числа по данной с. п. переводятся по одному — число, подлежащее переводу, нужно заслать в ячейку 0002.

Результат получается в ячейке 0003.

Например, печать массива $A[1 : n]$ реализуется в адресной записи следующим образом:

0006— i ;

0007— A ;

0010— n ;

3011—1 П А (единица второго адреса)

$a + 1$		3011		0006
$a + 2$	01	0007	0006	0002
$a + 3$	34	0002		0002
$a + 4$	30	3026	$a + 5$	3100
$a + 5$	22	0003	0003	$a + 6$
$a + 6$	01	0006	3011	0006
$a + 7$	04	0006	0010	$a + 2$
$a + 10$	33			

Все программы, изложенные в предыдущих главах на языке АЛГОЛ-60, преобразованы в рабочие программы с помощью рассмотренных выше правил перевода.

ЛИТЕРАТУРА

1. Бекус Дж. и др. Сообщение об алгоритмическом языке АЛГОЛ-60. «Журнал «Выч. мат. и мат. физики», 1961, № 2.
2. Битюцкий А. И. Некоторые вопросы расчета висячих покрытий с ортогональной пространственной сеткой нитей. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
3. Варвак П. М. Развитие и приложение метода сеток к расчету пластинок. К., изд-во АН УССР, ч. I, 1949, ч. II, 1952.
4. Варвак П. М. и др. Таблицы для расчета прямоугольных плит. К., изд-во АН УССР, 1959.
5. Глушков В. М., Ющенко Е. Л. Вычислительная машина «Киев». К., Гостехиздат УССР, 1962.
6. Глушков В. М. Универсальный алгоритмический язык АЛГОЛ-60. Материалы семинара по вопросам теории электронных цифровых машин, вып. 2. К., изд. Киевского дома научно-технической пропаганды, 1963.
7. Глушенко И. П. К расчету предварительно натянутых нитей с малыми стрелками. Львов, Научные записки Львовского политехнического ин-та, 1960.
8. Гнеденко Б. В., Королюк В. С., Ющенко Е. Л. Элементы программирования. М., Физматгиз, 1961.
9. Гордеев В. Н. Расчет висячих покрытий на некоторые виды нагрузок. Труды II-ой Всесоюзной конференции по пластинам и оболочкам. К., изд-во АН УССР, 1962.
10. Гордеев В., Перельмутер А., Скачков О. Расчет некоторых систем с односторонними связями. «Строительство и архитектура», 1962, № 9.
11. Гордеев В. Н. Уравнения для расчета тканевых оболочек (на украинском языке). «Прикладна механіка», вып. 6 т. VIII, 1962.
12. Гофман Ш. М. Расчет плоских рам способом последовательного уравнивания. Труды Ташкентского ин-та инженеров ж.-д. транспорта, 1962.
13. Гребень Е. С. Вопросы матричного расчета многократно статически неопределимых систем. Прочность и колебания конструкций. Вып. 172. Л., Сб. трудов ЛИИЖТ, 1960.
14. Длугач М. И. Обобщение метода Мора на определение перемещений в теории упругости. Сб. «Расчет пространственных конструкций» под ред. проф. А. А. Уманского. Вып. V, М., Госстройиздат, 1959.
15. Длугач М. И. Расчетная модель метода сеток (на украинском языке). «Прикладна механіка», 1962, вып. 3, т. II.
16. Длугач М. И., Шинкарь А. И. Применение электронных вы-

- числительных машин к решению бигармонической задачи (на украинском языке). «Прикладна механіка», 1962, вып. 2, т. VIII.
- 17. Дмитриев Л. Г. К расчету вантовых покрытий (на украинском языке). «Вісник АБІА УРСР», 1961, № 2.
 - 18. Дмитриев Л. Г., Городецкий А. С. Висячие покрытия. «Строительство и архитектура», 1962, № 7.
 - 19. Дмитриев Л. Г. Расчет висячих покрытий. «Строительство и архитектура», 1963, № 2.
 - 20. Ершов А. П., Кожухин Г. И., Волошин Ю. М. Входной язык системы автоматического программирования (предварительное сообщение). Сиб. отд. АН СССР, Институт математики. М., изд. Вычислительного центра АН СССР, 1961.
 - 21. Жукова Т. Р. Системы гибких нитей. «Строительная механика и расчет сооружений», 1962, № 1.
 - 22. Загорянский Л. А. Практический способ ортогональных предварительно напряженных вантовых сеток. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
 - 23. Качурин В. К. Гибкие нити с малыми прогибами. М., Гостехиздат, 1956.
 - 24. Качурин В. К. Теория висячих систем. М., Госстройиздат, 1962.
 - 25. Качурин В. К. О расчете круглого висячего покрытия с радиальным расположением вант. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
 - 26. Китов А. И. и Криницкий Н. А. Электронные цифровые машины и программирование. М., Физматгиз, 1959.
 - 27. Кирсанов Н. М. Вопросы проектирования висячих покрытий. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
 - 28. Кольчицкий М. С. Применение покрытий висячего типа в железобетонных резервуарах. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
 - 29. Кузнецов Э. Н. Некоторые вопросы расчета вантовых покрытий. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
 - 30. Кузнецов Э. Н. Статический расчет двухпоясных вантовых систем. «Строит. мех. и расчет сооруж.», 1962, № 5.
 - 31. Кузнецов Э. Н. Расчет висячих покрытий кругового очертания в плане. Труды НИИЖБ, вып. 2, М., ГСИ, 1962.
 - 32. Курочкин В. М. Компилирующая и интерпретирующая система использования стандартных программ для машины БЭСМ-2М. М., изд. Вычислительного центра АН СССР, 1961.
 - 33. Лавров С. С. Об экономии памяти в замкнутых операторных схемах. «Журнал выч. мат. и мат. физики», 1961, № 4.
 - 34. Лауль Х. Х., Кульбах В. Р. Из опыта моделирования и испытания висячего покрытия Таллинской певческой эстрады. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
 - 35. Людковский И. Г. Подвесные конструкции покрытий в СССР и за рубежом. «Архитектура СССР», 1960, № 1.
 - 36. Людковский И. Г. Современное состояние и перспективы применения висячих покрытий. Сб. «Висячие покрытия», М., Госстройиздат, 1962.
 - 37. Людковский И. Г. Характеристика вантовых систем и висячих оболочек и некоторые рекомендации по их проектированию. Труды НИИЖБ, вып. 2, М., ГСИ, 1962.
 - 38. Лесников В. В. Возникновение и развитие висячих покрытий кругового очертания в плане. Труды НИИЖБ, вып. 2, М., ГСИ, 1962.
 - 39. Магарик В. А. и Нагорный Н. М. Система команд универсальной цифровой автоматической машины БЭСМ-2. М., изд. вычислительного центра АН СССР, 1960.
 - 40. Мацелинский Р. Н. Статические расчеты гибких висячих конструкций. М., Госстройиздат, 1952.
 - 41. Мацелинский Р. Н. Статический расчет упругих нитей. «Строит. мех. и расчет сооруж.», 1959, № 4.

42. Нарец Л. К. Расчет статически неопределимых систем на малых вычислительных машинах. М., Гостройиздат, 1958.
43. Петров В. В. К расчету пологих оболочек при конечных прогибах. Доклады высшей школы. Раздел «Строительство», 1959, № 1.
44. Пухов Г. Е. Электрическое моделирование стержневых и тонкостенных конструкций. К., изд. АН УССР, 1960.
45. Пшеничников Г. И. К расчету висячих покрытий круглых в плане. «Инженерный журнал», вып. 3, т. 2, 1962.
46. Рабинович И. М. Курс строительной механики стержневых систем, ч. II, М., Гостройиздат, 1954.
47. Рабинович И. М. Мгновенно жесткие системы, их свойства и основы расчета. Сб. «Висячие покрытия». М., Гостройиздат, 1962.
48. Райнус Г. Э. Теория статического расчета некоторых типов висячих систем. Автореферат диссертации. Л., изд. АСИА СССР, 1963.
49. Резников Р. А. К решению задач строительной механики на электронных вычислительных машинах. «Строит. мех. и расчет сооруж.», 1959, № 4.
50. Резников Р. А. Об автоматизации расчетов строительных конструкций. Сб. «Механизация инженерно-технических расчетов при проектировании сооруж.». М., Гостройиздат, 1959.
51. Рогоцкий С. А. Расчет рам. М., Машгиз, 1948.
52. Силицын А. П. Расчет сеток. «Вестник Военно-инженерн. академии им. Куйбышева», Сб. по строит. мех., 1936.
53. Силицын А. П. Упрощенный расчет сеток. «Вестник Военно-инженерной академии им. Куйбышева», Сб. по строит. мех., 1941.
54. Современные методы расчета сложных статических неопределимых систем. Сборник статей иностранных авторов. Составление и общая ред. проф. А. П. Филина. М.-Л., 1961.
55. Смирнов А. Ф. Устойчивость и колебания сооружений. М., Трансжелдориздат, 1958.
56. Смирнов А. Ф. Задачи строительной механики в связи с применением вычислительной техники. «Строит. мех. и расчет сооруж.», 1963, № 1.
57. Сосис П. М. Расчет рам способом перераспределения начальных значений неизвестных. Изд. 2-ое, дополненное, К., Гостехиздат, 1956.
58. Сосис П. М. Решение n -членных уравнений строительной механики методом перераспределения начальных значений неизвестных (на украинском языке). «Прикладна механіка» вып. 4, т. V, 1959.
59. Сосис П. М. Опыт механизации инженерных расчетов на машинно-счетной станции. «Строит. мех. и расчет сооруж.», 1960, № 3.
60. Сосис П. М. Метод составления программ инженерных расчетов. Сб. «Исследования по теории сооруж.», вып. X, М., Гостройиздат, 1961.
61. Сосис П. М. Механизация расчетов сооружений по типовым программам. К., Гостройиздат, 1961.
62. Сосис П. М., Дмитриев Л. Г. Программирование и применение электронных вычислительных машин для расчета строительных конструкций. Труды VI сессии Академии строительства и архитектуры УССР. Киев, 1962.
63. Сосис П. М. О методах решения систем линейных уравнений с малозаполненными матрицами коэффициентов на электронных цифровых машинах, Сб. «Исследования по теории сооружений», вып. XII, М., Гостройиздат, 1963.
64. Строительная механика в СССР (1917—1957), под ред. чл.-корр АН СССР И. М. Рабиновича. М., Гостройиздат, 1957.
65. Тарнопольский Б. Л. Статические расчеты многоэтажных и многопролетных рам на электронной цифровой машине. Сб. «Механизация инженерно-технических расчетов при проектировании сооружений». М., Гостройиздат, 1959.

66. Уманский А. А. Пространственные системы. М., Госстройиздат, 1948.
67. Фадеев Д. К., Фадеева В. Н. Вычислительные методы линейной алгебры. М., Физматгиз, 1960.
68. Филин А. П. Расчет пространственных стержневых конструкций типа системы перекрестных связей и его применение к оболочкам при использовании электронных вычислительных машин. Исследования по строительной механике. Сб. трудов ЛИИЖТ, вып. 190, Л., 1962.
69. Фридман Л. Б. Висячие армоцементные покрытия киноконцертного зала в Харькове. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
70. Харитонов Е. П. Экспериментальное исследование висячих предварительно напряженных оболочек кругового очертания в плане. Сб. «Висячие покрытия». М., Госстройиздат, 1962.
71. Цаплин С. Теория расчета гибких нитей. М., Всесоюзная строительная выставка, 1937.
72. Чаусов Н. С. Применение электронных вычислительных машин при расчете инженерных сооружений. М., Госстройиздат, 1962.
73. Шайкевич В. Д. Матричный метод расчета регулярных систем. Сб. «Расчет пространственных конструкций», под ред. проф. А. А. Уманского. Вып. VI. М., Госстройиздат, 1958.
74. Rivlin R. S. Plane strain of a net formed by inextensible cords. «Rational Mech. and Analysis», 1955, № 6.
75. Bandel H. K. Das ortogonale Seilnetz hiperbolisch parabolischer Form unter vertikalen Lastzuständen und Temperaturänderung. «Bauingenier», 1959, № 10.
76. Eras, Elze. Berechnungsverfahren für vorgespannte, doppelt gekrümmte Seilnetzwerke. Bauplanung. «Bautechnik», 1961, № 7.
77. Сборник «Механизация и автоматизация расчетов в строительном проектировании (из опыта работы проектных организаций Госстроя СССР)». Изд. Гипротис Госстроя СССР и Оргкомитета Всероссийского научно-технического совещания по механизации и автоматизации инженерных и управленческих работ в промышленности и строительстве. М., 1963.
-

ОГЛАВЛЕНИЕ

	Стр.
Предисловие	3
Глава I. О новом подходе к методам расчета в связи с развитием вычислительной техники и программирования	5
Глава II. Обозначения (языки) в системах программирования	16
Инженерное программирование	16
Адресное программирование	32
Программирование на языке АЛГОЛ-60	45
Сравнение систем программирования и предложения по их улучшению	63
Глава III. О решении систем линейных уравнений строительной механики на ЭВМ	75
Использование особенностей структур матриц для повышения возможного порядка систем, решаемых на ЭВМ	75
Оптимальные программы решения систем линейных уравнений методом исключения	84
Глава IV. Пологие пространственные конструкции	94
Примеры из практики проектирования, анализ расчетных схем	94
Проблема статического расчета пологих пространственных конструкций	113
Обоснование принятых методов расчетов	119
Глава V. Вывод основных уравнений	124
Глава VI. Расчет пологих пространственных систем в нелинейной постановке (вантовые сети)	141
Глава VII. Обобщенные программы	150
Глава VIII. Применение обобщенных программ	163
Глава IX. Оптимальные программы расчетов сооружений	191
Приемы организации памяти ЭВМ при расчете статически неопределимых систем	191
Оптимальная программа расчета вантовых систем методом многоступенчатого нагружения	204
Глава X. Перевод программ с языка АЛГОЛ-60 на язык ЭВМ	212
Литература	223

Зональный научно-исследовательский и проектный институт типового
и экспериментального проектирования жилых и общественных зданий
Госкомитета по гражданскому строительству и архитектуре при Госстрое СССР

Леонид Георгиевич Дмитриев,

Петр Моисеевич Сосис

Программирование расчета пространственных конструкций

Редактор А. С. Гончар

Технический редактор Н. Л. Леущенко

Корректор А. А. Луковская

БФ 34144. Сдано в набор 23.VIII-1963 г. Подписано к печати 2.XII-1963 г.
Бумага $60 \times 90^{1/16} = 7,125$ бумажных, 14,25 физ. и усл. печ., 12,99 уч.-изд. листов.
Тираж 4000. Цена 75 коп. Темплан 1963 г., № 70 Зак. 1313.
Государственное издательство литературы по строительству и архитектуре УССР
Киев, Владимирская, 24

Типография Госстройиздата УССР, Киев, Выборгская, 84