ИНСТИТУТ ЯДЕРНОЙ ФИЗИКИ
им. Г.И. Будкера СО РАН

В.Н. Иванченко

СОСНА—ПАКЕТ ПРОГРАММ ДЛЯ СТРУКТУРИРОВАНИЯ ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ

ИЯФ 94-25

НОВОСИБИРСК

СОСНА — пакет программ для структурирования экспериментальных данных

В.Н.Иванченко

Институт ядерной физики
630090, Новосибирск, Россия

Аннотация

Приводится полное описание пакета программ СОСНА, который предназначен для обеспечения разработки программ реконструкции событий в экспериментах по физике частиц высоких энергий. СОСНА позволяет создать структуру данных, описывающих событие, в рамках модели объектов и отношений. Проводится сравнение с известными пакетами программ, обсуждается опыт использования пакета СОСНА для создания программы реконструкции событий с детектора СНД.

Содержание

1 Введение

Уже в течение многих лет при подготовке математического обеспечения экспериментов по физике высоких энергий используются пакеты программ для работы со структурами данных, наиболее известные из них ADAMO [1], BOS [2], JAZELLE [3], TYPES [4], ZEBRA [5] и др. Эти пакеты обеспечивают возможность создание программ полной реконструкции событий в больших современных детекторах многочисленным коллективам авторов, часто разбросанных по разным странам. Унификация структур данных позволяет реализовать четкий протокол связи между разработчиками отдельных частей программы, удобно структурировать программу одного разработчика, обеспечить возможность разработки программ в различных операционных системах, что в совокупности позволяет гарантировать качество математического обеспечения для сложного детектора.

Пакет программ СОСНА создавался в Новосибирске в рамках подготовки к экспериментам со Сферическим нейтральным детектором (СНД) на e^+e^- коллайдере ВЭПП-2М и на будущей Ф-фабрике [6]. При разработке этого пакета программ учитывались следующие особенности экспериментов в Новосибирске: сравнительная простота детекторов, высокий ожидаемый поток данных, ограниченные компьютерные ресурсы, постоянные изменения компьютерного парка в Новосибирске. СОСНА отличается компактностью, минимальными накладными расходами при

работе со структурами данных, независимостью от каких-либо других пакетов программ.

В данной работе в параграфе 2 приводятся краткий обзор существующих программ, поддерживающих структуры данных. Концепция пакета программ СОСНА находится в параграфе 3. В параграфах 4.–4.6 приводится полное описание пакета программ СОСНА с детальным обсуждением его особенностей. Список общих областей — в параграфе 5. Список всех подпрограмм находится в параграфе 6. Пример использования СОСНы для реконструкции событий детектора СНД обсуждается в параграфе 7.

2 Модель объектов и отношений

В течение ряда лет в физике высоких энергий имел место заметный прогресс. Техника эксперимента непрерывно совершенствовалась, детекторы частиц усложнялись. Естественно, что совершенствовались и методы обработки данных. Причем уже давно математическое обеспечение эксперимента готовится коллективом, а для крупных современных детекторов и несколькими коллективами авторов. В связи с этим совершенно необходимыми оказались формализация и структурирование данных, обеспечение стандартного протокола для взаимодействия различных частей программ обработки. Ключевым моментом для такой формализации явилось понимание необходимости разделения данных, описывающих событие, и собственно математических алгоритмов. Опыт многих экспериментов показал, что для представления данных о каждом событии в различных детекторах могут быть использованы одинаковые структуры данных, при том, что математические алгоритмы преобразования данных для каждого детектора индивидуальны.

При создании структур данных необходимо учитывать, что в каждом отдельном событии рождается разное число частиц, и происходит срабатывание разного количества элементов детектора. Данными, описывающими событие являются не только коды сработавших счетчиков, камер и других элементов детектора, но и восстановленные треки в камерах, кластеры в калориметрах и другие объекты, полученные в результате обработки. Треки составляются из сигналов камер, кластеры — из срабатываний башен калориметра, треки и кластеры собираются вместе в заряженные частицы. Некоторые кластеры оказываются не связанными с треками, они являются кластерами от нейтральных частиц, например, фотонов.

Структуру данных можно формализовать, разделив их на два класса, которые принято называть объектами и отношениями (entity-relashionship model of data [7]). Набором объектов называется совокупность однотипных объектов, например, треков, кластеров, сработавших проволочек. Каждый объект из набора описывается некоторым фиксированным числом параметров объекта. Объекты из одного набора могут быть связаны с объектами другого набора, например, проволочки в камере собираются в трек. Структура данных, описывающая связи между объектами, называется отношением. Отношение связывает один объект из некоторого набора объектов с одним или несколькими объектами другого набора. Таким образом, отношение имеет направленность. Кроме того отношение может иметь параметры, например, отклонение координаты проволочки от трека в дрейфовой камере можно рассматривать в качестве параметра отношения "проволочка→трек". Отметим, что в существующих пакетах программ реализованы только отношения типа "один объект→один объект" или "много объектов→один объект", а более сложные конструкции отношений получаются с использованием этих базовых структур.

Модель объектов и отношений позволяет создать графическое представление программы реконструкции событий в виде диаграммы. Пример такой диаграммы для пакета программ ADAMO [1] приведен на рис. 1. На рисунке прямоугольниками показываются наборы объектов, стрелками — отношения между ними, внутри прямоугольников приведены списки параметров объектов. Графическая диаграмма позволяет разработать и наглядно представить полный проект программы реконструкции событий, не приступая к его реализации, выбрать соответствующие алгоритмы обработки. После создания графической диаграммы определяется структура данных для данного детектора, в то же время автоматически структурируется будущая программа реконструкции событий. Отдельные части диаграммы соответствуют рекострукции данных отдельными подсистемами детектора и могут быть реализованы различными группами авторов, а связи между ними формализованы зафиксированными на диаграмме наборами объектов и отношений между ними.

Кратко охарактеризуем некоторые существующие пакеты программ для работы со структурами данных. Одним из старейших из них является BOS [2], который был создан в DESY и до сих пор используется в CERN и в DESY (ALEPH, CDF, H1). Структура данных BOS создается и поддерживается в одной общей области программы, она состоит из набора именованых банков, структура каждого определяется пользо-

вателем. Явного определения отношений и их параметров нет. Для этого необходимо использовать дополнительные пакеты программ или хранить параметры отношений в специально созданных банках BOS.

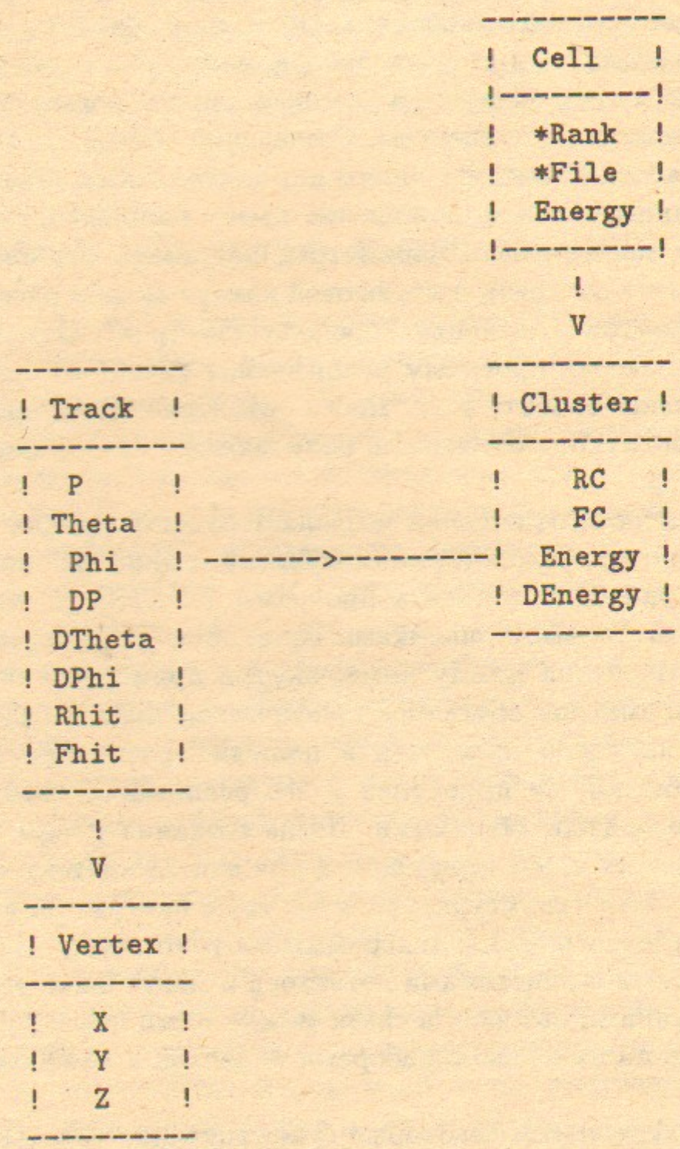

Рис. 1. Пример диаграммы объектов и отношений в ADAMO [1]. Прямоугольники — наборы объектов, в верхней строке имена наборов, ниже приведены списки имен параметров каждого объекта из данного набора. Стрелками показаны отношения между наборами.

Наиболее широкое распространение в настоящее время получил пакет программ ZEBRA [5], который был разработан для обеспечения экспериментов на LEP (ALEPH, L3, OPAL), но был принят в качестве базового пакета для многих других экспериментов (D0, ZEUS, ...). ZEBRA используется очень широко, являясь базой для программы моделирования GEANT, графических и других пакетов программ, разработанных в CERN. Структуру данных ZEBRA можно описать в терминах модели объектов и отношений, но по своей сути эта структура организована в виде сети. В каждом узле сети находится банк данных с параметрами одного объекта и информацией об отношениях с другими объектами (рис. 2). Узлы набора однотипных объектов связаны между собой последовательно взаимнооднозначными связями (next на рис. 2) и образуют линейную структуру. Объекты разного сорта связаны между собой структурными связями (down на Рис.2), для обеспечения которых пользователь должен выделить место в банках объектов. Большим достоинством структуры ZEBRA является возможность добавления, модификации и уничтожения объектов в произвольном порядке. Возможно создание одновременно нескольких совершенно независимых структур данных.

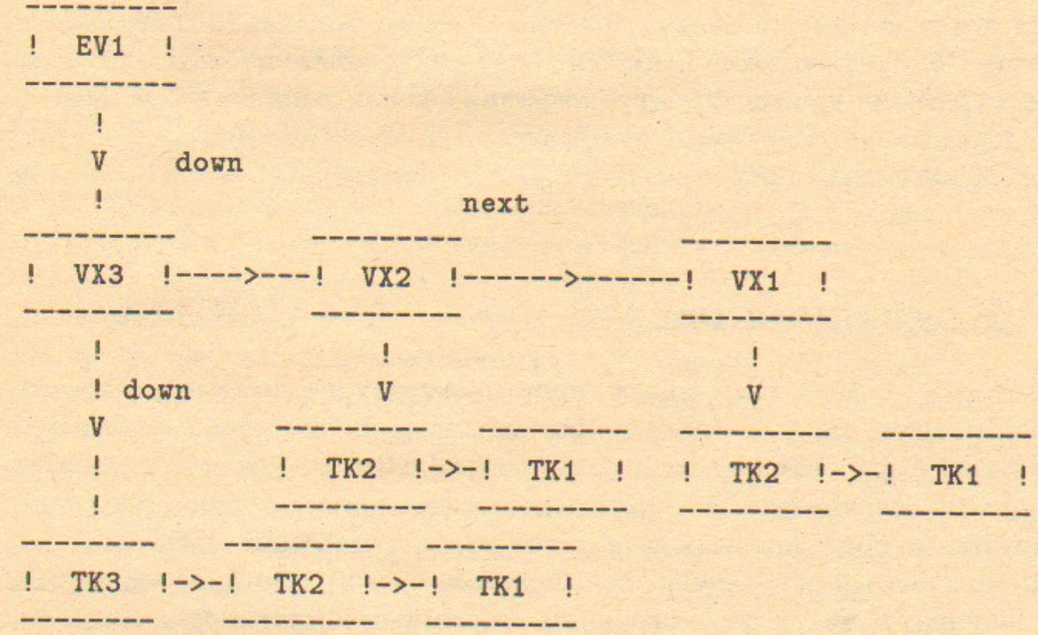

Рис. 2. Пример структуры данных ZEBRA [5]. Прямоугольниками показаны банки с параметрами объектов, горизонтальные линии — линейные связи однотипных объектов, вертикальные линии — структурные связи между объектами различных сортов.

В CERN был также разработан пакет программ ADAMO [1], в котором последовательно реализована модель объектов и отношений, этот пакет используется в экспериментах ALEPH и ZEUS. Пакет программ ADAMO имеет два уровня. Пользователь работает с верхним уровнем, он должен вначале определить структуру своих данных в текстовых файлах, затем специальными командами преобразовать их в другие файлы, которые вызываются при трансляции подпрограмм обработки данных и обеспечивают создание системы окон в памяти в виде общих областей, к которым пользователь имеет прямой доступ. Пользователь имеет возможность создавать, модифицировать, уничтожать объекты, есть возможность осуществлять навигацию по структуре данных. ADAMO поддерживает систему окон, используя пакет подпрограмм нижнего уровня, сохраняя и модифицируя банки с данными в некоторой базовой структуре данных. На первом этапе разработки ADAMO предполагалось, что данные на нижнем уровне будут храниться в базе данных ORACLE, однако в дальнейшем оказалось оптимальным использовать банки BOS или ZEBRA. Наряду со многими достоинствами существующих пакетов программ поддержки структур данных, следует отметить, что они оказываются достаточно громозкими, правила их использования нетривиальны. Поэтому осуществляются попытки реализовать новые подходы, которые при сохранении уровня структурирования данных были бы более просты в использовании. Например, для возможных экспериментов на SSC разрабатывался пакет программ TYPES [4], в Новосибирске был разработан пакет программ COCHA.

3 Концепция пакета программ COCHA

Пакет программ COCHA сориентирован на создание системы обработки экспериментов на ф-фабрике в Новосибирске. Если сравнить эксперименты на ф-фабрике с экспериментами на LEP в CERN или с другими крупными современными экспериментами, то главное отличие состоит в значительно более простых детекторах и, соответственно, событиях. Однако ожидаемый поток событий с Ф-фабрики очень велик, по-видимому, значительную часть реконструкции событий необходимо будет выполнять уже в реальном времени, до решения о записи события в выходной файл. Таким образом, структура данных должна быть создана в программе, работающей в реальном времени. Это одна из причин, по которой представляется оптимальным разработка оригинального пакета

программ COCHA для создания структур данных, не связанного с каким-либо другим пакетом программ.

При его создании также учитывался реально существующий компьютерный парк и прогнозировалось его возможное развитие. Основные соображения при разработке пакета программ COCHA состояли в следующем:

— создаются отдельные структуры данных для наборов объектов и для отношений, что позволяет работать с отношениями отдельно, в том числе, реализовать алгебру отношений;

— все объекты, отношения и их параметры именованы;

— параметры объектов и отношений имеют длину 4 байта и могут быть целыми, вещественными или текстовыми;

— создание, модификация и доступ к объектам и отношениям сопровождается минимумом накладных расходов, как правило, доступ к банку с параметрами объекта или отношения состоит в передаче адреса;

— набор объектов находится в банке данных в буфере COCHы в виде непрерывного двумерного массива, каждый элемент отношения хранится независимо, и отношение в целом не обязательно занимает непрерывную область памяти;

— параметры каждого набора объектов или отношения описывается в одном месте, т.е. в одной подпрограмме, изменение одного описания не влияет на описание или доступ к другим описаниям;

— наборы объектов делятся на два класса — постоянные и обновляемые, последние изменяются на каждом событии;

— при записи или чтении события из файла параметры события упаковываются в целые числа длиной 2 байта;

Главное отличие пакета COCHA от известных импортных пакетов состоит в том, что для создания структур данных по наборам объектов с одной стороны и по отношениям между ними используются разные буфера со структурами данных, что обеспечивает более гибкую работу. В первую очередь, это позволило реализовать алгебру отношений, т.е. складывать, умножать, обращать отношения. Кроме того, это заметно облегчает саму структуру данных, соответственно упрощает обслуживающие программы, делает программу более эффективной.

В пакете COCHA несколько упрощено задание имен. Имя набора объектов или отношения состоит из 8 символов, если пользователь задал меньше, то имя справа дополняется пробелами. Имена параметров все длиной 4 байта. Также ограничен выбор форматов переменных, все они имеют длину 4 байта.

Параллельно с пакетом программ СОСНА разработан специфический формат записи событий в файл, который будет храниться на магнитных накопителях, все данные упаковываются в целые переменные длиной 2 байта. Такой формат позволяет максимально сжать информацию в файле с данными, не потеряв точности, это особенно важно для условий в Новосибирске, когда объем доступной памяти для хранения файлов с данными ограничен.

Пакет программ СОСНА написан на языке ФОРТРАН-77, перевод пакета с операционной системы VM на компьютерах типа IBM к операционной системе VAX VMS прошел в свое время без каких-либо изменений в текстах программ. Следует ожидать, что пакет программ также без существенных переделок может быть запущен на ферме из спецпроцессоров, что позволит включить программу полной реконструкции событий в режиме реального времени.

Важная особенность пакета программ СОСНА состоит в том, что в нем имеются одновременно подпрограммы обеспечивающие нижний и верхний уровни программы. Наличие ряда ограничений несущественно для сравнительно малых экспериментов и компенсируется простотой использования.

4 Описание пакета программ СОСНА

Все имена основного пакета подпрограмм имеют длину 6 символов и начинаются с символов SC13, подпрограммы, обслуживающие ввод-вывод событий на внешние магнитные накопители, имеют 6-символьные имена, начинающиеся с символов SR13, SW13 или SP13. Имена всех общих областей начинаются с символов SND.

При неправильно заданных параметрах подпрограмм пакета, которые могут повлечь за собой нарушения в структуре данных, выдается сигнальное сообщение, и нештатная ситуация сглаживается. Как правило, подпрограмма просто не выполняет никаких действий. Сообщение подпрограмм СОСНы о неправильно заданных параметрах обязательно содержит цепочку вызовов подпрограмм. В ИЯФ СО РАН пакет программ СОСНА находится в библиотечном файле

```
disk$d5:[ivanchenko.snd_lib]cocha_v2.olb
```

Как правило, при использовании СОСНы не требуется специальной инициализации, поскольку эта инициализация происходит при первом обращении к любой из подпрограмм пакета. Однако может оказаться, что

какой-либо буфер со структурой данных имеет недостаточную длину. В этом случае пользователю понадобится его удлинить. Тогда он может инициализировать пакет с помощь вызова двух подпрограмм

```
call SC1311(0)
call SC1314('/SND3##/',L)
```

Здесь символами ## обозначен номер общей области, которую необходимо удлинить (список общих областей см. в п.5), L — новая максимальная длина общей области. Естественно, что пользователь должен описать эту общую область длины L в своей подпрограмме.

В следующих параграфах будут обсуждаться варианты работы с пакетом СОСНА. В этом описании все выходные параметры подпрограмм будут помечены символом *, по умолчанию предполагается ФОРТРАН определение переменных INTEGER*4 и REAL*4. Все адреса в структуре данных СОСНы — это индексы элементов массивов в общих областях, которые описаны в п.5. Термин “указатель на начало банка” в данном описании означает индекс первого элемента банка минус единица.

4.1 Инициализация объектов и отношений

Любой набор объектов и любое отношение должны быть зарегистрированы в СОСНе. Регистрация может осуществляться в любом месте программы пользователя. Представляется удобным каждому из разработчиков группы, работающей над созданием общей программы, зарегистрировать свои наборы объектов и свои отношения в одной своей подпрограмме, которая вызывается в начале счета. Регистрация объекта осуществляется вызовом подпрограммы

```
call SC1300(NAME,L,NAMP,ID*)
```

NAME — (CHARACTER*8) имя набора, L — число параметров объекта набора, NAMP(L) — (CHARACTER*4) массив имен параметров объекта, ID — идентификатор набора, который одновременно является номером набора. Большинство подпрограмм пакета работает с идентификаторами, часть—с именами наборов. В любой из подпрограмм пользователя идентификатор ранее зарегистрированного набора ID и число параметров объекта L в этом наборе можно узнать по имени

```
call SC1301(NAME,ID*,L*)
```

Удобно данный вызов осуществлять только один раз при первом входе в подпрограмму пользователя.

Отношение между наборами объектов можно зарегистрировать, если уже определены идентификаторы наборов ID1 и ID2

```
call SC1304(NAME,ID1,ID2,L,NAMP,ID*)
```

Смысл параметров подпрограммы такой же, как и у SC1300, ID — идентификатор (номер) отношения. В отличие от объектов у отношения число параметров L может быть равно нулю. Отношение имеет направленность ID1→ID2, то есть показывает какие дочерние объекты из набора ID1 имеют отношение к каждому материнскому объекту из набора ID2. В любой из подпрограмм идентификатор ранее зарегистрированного отношения ID и число параметров отношения L можно узнать по имени

```
call SC1305(NAME,ID*,L*)
```

После обработки одного события перед обработкой следующего необходимо инициализировать структуру данных. Это осуществляется с помощью подпрограммы

```
call SC1311(ID)
```

где ID — идентификатор (номер) последнего набора, который не должен изменяться перед обработкой следующего события. Таким образом наборы объектов делятся на два класса:

— наборы объектов с номерами не большими ID, которые заполняются один раз в данном счете и остаются неизменными, в таком виде удобно хранить константы обработки;

— наборы объектов обновляемые на каждом событии, содержащие собственно параметры события.

Следует отметить, что при описываемой инициализации перед началом обработки следующего события наборы объектов и отношения не удаляются из структуры, а происходит всего лишь зануление количества объектов в каждом наборе и, соответственно, во всех отношениях. Это означает, что нет необходимости в повторной регистрации наборов и отношений между ними.

4.2 Заполнение объектов и отношений

Существует несколько способов заполнения банков данных с параметрами объектов. В любом случае перед заполнением набор должен быть

зарегистрирован. Один из вариантов записи состоит в том, что в буфере пользователя формируется строка с параметрами очередного объекта, а затем осуществляется ее сохранение

```
call SC1302(ID,A)
```

ID — идентификатор набора, A — адрес начала строки. Возможен вариант, когда пользователем формируется сразу буфер с параметрами K объектов

```
call SC1325(ID,K,A)
```

Если формирование одного набора объектов осуществляется последовательно, то накладные расходы при таком способе формирования набора связаны только с передачей строки из буфера пользователя в банк данных СОСНы. Если одновременно создаются несколько наборов, то накладные расходы могут заметно возрасти. Уменьшить накладные расходы можно, если предварительно заказать необходимый буфер в банке данных СОСНы

```
call SC1324(ID,K,IA*)
```

K — количество дополнительных объектов набора ID, для которых выделяется место под банк с K объектами в буфере общей области /SND321/ и прописывается нулями, IA — указатель на начало этого банка (адрес — 1). Следует отметить, что в буфере объектов СОСНы банк, содержащий набор объектов, расположен непрерывно. Следовательно, если в наборе ID уже были записаны объекты, то дополнительный банк, запрошенный через подпрограмму SC1324, будет располагаться вслед за ранее сформированным банком, составляя единое целое. После подпрограммы SC1324 можно прямо заполнять банк, либо воспользоваться подпрограммой SC1319.

Пожалуй наиболее эффективный способ работы со структурой СОСНы обеспечивает подпрограмма SC1319, обращение к которой следующее:

```
call SC1319(User_code,N,ID1,ID2,ID3,ID4,ID5,ID6)
```

User_code — имя подпрограммы пользователя, которое необходимо описывать в вызывающей программе в операторе EXTERNAL; N — количество наборов объектов, используемых в User_code ($N < 7$); ID1#6 — их идентификаторы. Причем, в вызывающей программе можно ограничиться заданием N идентификаторов, если $N < 6$ (см. пример п.7). Подпрограмма User_code должна быть оформлена по следующим правилам:

```
Subroutine User_code(B1,IB1,L1,K1,B2,IB2,L2,K2,....)
Dimension B1(L1,1),IB1(L1,1),B2(L1,1),IB2(L1,1)....
```

Число пар массивов B# и IB# определяется параметром N — это число наборов объектов необходимых пользователю. Эти пары массивов находятся в одной области памяти, т.е. эквивалентны по правилам ФОРТРАНа. Удобство работы через вызов подпрограммы SC1319 состоит в том, что пользователь имеет дело в подпрограмме User_code с двумерным массивом, а не с адресом в одномерном массиве. Плата за это удобство состоит в ряде ограничений:

— число наборов объектов не превышает 6 (автор пакета СОСНА может увеличить это число при необходимости в новой версии пакета);

— в одной подпрограмме, вызваной таким образом, может быть создан только один набор объектов, т.е. только у одного набора объектов величина K# при входе равна 0, а при выходе > 0;

— запрещено увеличивать число объектов в наборе, однако нет ограничений на уменьшение числа объектов в наборе, т.е. если K# на входе было больше 0, то оно может быть уменьшено внутри User_code, но не может быть увеличено;

— внутри подпрограммы нельзя вызывать подпрограммы SC1302, SC1324, SC1325 с идентификаторами ID1#6, описанными в данном вызове SC1319.

С помощью специальной проверки было показано, что использование SC1319 на VAX кластере ИЯФ СО РАН обеспечивает создание наиболее быстрой программы по сравнению с другими вариантами работы с пакетом СОСНА, кроме того текст такой программы наиболее легко читается. Описанные правила с одной стороны ограничивают пользователя, с другой стороны его дисциплинируют. Требование создавать внутри одного вызова SC1319 только один набор объектов вынуждают его составить диаграмму объектов и отношений и продумать непротиворечивую и достаточно ясную стратегию реконструкции событий, максимально структурируя программу.

В отличие от наборов объектов, отношения могут создаваться и заполняться только одним способом

```
call SC1306(ID,K2,L,IB,K2M,IBP)
```

ID — идентификатор отношения, K2 — номер материнского объекта, L — длина списка дочерних объектов, IB — буфер пользователя со списком дочерних объектов, K2M — максимально возможный номер материнского

объекта, IBP — буфер пользователя с параметрами отношения. Параметр K2M реально используется только при первом занесении отношения для данного события. При этом полное число объектов материнского набора может быть еще неизвестно, поэтому пользователю каким-либо образом необходимо определить максимальное возможное значение K2, может быть даже со значительным запасом.

При создании наборов объектов и при дальнейших манипуляциях с ними бывает необходимо использовать временные рабочие массивы переменной длины. Для этой цели в СОСНе имеется возможность использовать буфер имен в общей области /SND322/. Пользователь запрашивает себе массив длины L с помощью вызова

```
call SC1320(L,IA*)
```

IA — адрес начала массива.

4.3 Доступ к объектам и отношениям

Самый эффективный способ доступа к параметрам набора объектов заключается в использовании подпрограмы SC1319, которая описана в предыдущем параграфе. Существуют и другие возможности, в частности,

```
call SC1316(ID,K*,IA*)
```

K — количество объектов в наборе, IA — указатель (адрес — 1) на начало банка с первым объектом набора в буфере объектов /SND321/.

Пользователь также может получить банк с объектами набора в свой буфер A, используя следующую подпрограмму:

```
call SC1303(ID,LM,A*,K*)
```

LM — максимальная длина буфера A. Следует отметить, что последняя возможность самая неоптимальная, поскольку происходит дополнительная пересылка массива. Кроме того, расположение параметров объектов в буфере пользователя будет зависеть от числа параметров объектов L, изменения L в программе инициализации нужно сопровождать соответствующими изменениями в подпрограммах пользователя, использующих описываемый вариант доступа.

Доступ к отношению осуществляется следующим образом:

```
call SC1307(ID,K2,L*,IA*,IAP*,*M)
```

K2 — номер материнского объекта, L — длина списка дочерних объектов, IA — указатель (адрес — 1) на начало списка дочерних объектов в буфере отношений /SND325/, IAP — указатель (адрес — 1) на начало банка с параметрами отношения в буфере отношений /SND325/, M — метка выхода, если список пуст.

Если пользователь хочет получить отношение в свои буфера, то может воспользоваться вызовом подпрограммы

```
call SC1317(ID,K2,LM,IA,LMP,AP,L*,*M)
```

LM — максимальный размер буфера пользователя IA для списка дочерних объектов, LMP — максимальный размер буфера пользователя AP для параметров отношения. Последний способ работы с отношениями неоптимален.

4.4 Модификация объектов и отношений

Самый эффективный способ модификации параметров набора объектов заключается в использовании подпрограмы SC1319, которая описана в параграфе 4.2. Возможно использовать также подпрограмму SC1316 для определения адреса в буфере объектов общей области /SND321/, а затем осуществить модификацию параметров, в этом случае пользователь должен сам побеспокоиться о том, чтобы не разрушить структуру данных. Если в программе пользователя оказался подготовлен новый буфер с параметрами объекта, то он может быть внесен в имеющуюся структуру

```
call SC1328(ID,K,A)
```

K — номер объекта в наборе, A — буфер пользователя.

Если пользователю необходимо удалить объект с номером K из набора ID, то можно воспользоваться подпрограммой

```
call SC1326(ID,K)
```

Причем, удаление последнего объекта из набора сопровождается наименьшими накладными расходами. Следует отметить, что удаление не последнего объекта из набора как с помощью подпрограммы SC1326 так и внутри подпрограммы User_code, обращение к которой организовано с помощью подпрограммы SC1319, нарушает структуру отношений этого набора объектов. Чтобы этого не происходило, для удаления объекта с номером K из набора ID следует вызвать подпрограмму

```
call SC1329(ID,K,NM,IDM,ND,IDD)
```


NM — число отношений, для которых данный набор является материнским, IDM(NM) — массив идентификаторов этих отношений, ND — число отношений, для которых данный набор является дочерним, IDD(ND) — массив идентификаторов этих отношений. Если NM=ND=0, то программа SC1329 работает аналогично программе SC1326.

В любом случае, удаление объектов — это неестественная процедура для пакета программ СОСНА. Гораздо более естественно для СОСНы является создание иерархии объектов разного уровня и системы отношений между ними. Например, при поиске кластеров в калориметре может быть создан набор "предварительные кластеры" (1-й уровень), определены параметры объектов набора, определены параметры треков в координатной системе, а уже после этого может быть уточнен список окончательных кластеров. Для этого создается другой набор объектов "окончательные кластеры" (2-й уровень). При этом никакие объекты не удаляются из наборов. В таком методе работы есть определеный плюс, заключающийся в том, что в структуре данных сохраняется вся промежуточная информация, что важно при отладке программы пользователя.

После вызова подпрограммы SC1307 (см. п. 4.3) пользователю доступны банки данных отношения, содержащие списки дочерних объектов и параметры отношения. Пользователь может их модифицировать, самостоятельно заботясь о том, чтобы не разрушить структруру данных. Если после обработки ему необходимо уменьшить длину списка, то можно воспользоваться вызовом

```
call SC1327(ID,K2,L1)
```

L1 — новая длина (L1«L), допустимо, чтобы L1=0.

Если необходимо изменить лишь параметры отношения ID, не изменяя списка дочерних объектов, относящихся к материнскому объекту с номером K2, то можно воспользоваться подпрограммой

```
call SC1318(ID,K2,A)
```

A — буфер пользователя с новыми параметрами отношения.

4.5 Алгебра отношений

По ходу обработки данных бывает полезно использовать отношения между наборами объектов, расположенных далеко друг от друга на диаграмме объектов и отношений. Например, можно заинтересоваться тем, какие башни калориметра принадлежат данной частице, тогда как из

башен строились кластеры и только затем из кластеров и треков в координатной системе были получены частицы. Может возникнуть также и другой вопрос: какие частицы имеют отношение к данной башне. Все вопросы такого рода удобно решать с помощью подпрограмм, обеспечивающих алгебру отношений.

Обращение отношения с идентификатором ID12 "набор1→набор2" можно осуществить подпрограммой

```
call SC1308(ID12,ID21)
```

ID21 — идентификатор обратного отношения "набор2→набор1". Отметим, что все параметры отношения при обращении сохраняются, если число параметров обращенного отношения не меньше исходного (L21>L12), иначе в отношение ID21 передается L21 первых параметров отношения ID12.

Произведение отношений ID12 ("набор1→набор2") и ID23 ("набор2→набор3") реализуется подпрограммой

```
call SC1309(ID12,ID23,ID13)
```

ID13 — идентификатор отношения "набор1→набор3". Параметры отношения сохраняются, если число параметров полученного отношения L13>L12+L23, иначе часть параметров не передается конечному отношению ID13.

Возможно также сложение двух различных отношений ID1 и ID2, связывающих одну и ту же пару наборов объектов.

```
call SC1310(ID1,ID2,ID)
```

ID — идентификатор суммарного отношения. В этой подпрограмме передача параметров отношений не проводится.

4.6 Вывод объектов и отношений

Вывод распечаток объектов и отношений в форматном виде в файл осуществляется с использованием канала вывода ФОРТРАН, заданного в переменной LUN в общей области /SND700/, по умолчанию номер канала равен 6.

Распечатка набора объектов ID возможна по столбцам

```
call SC1312(ID)
```


Если ID отсутствует в списке идентификаторов, то выводятся все наборы. При распечатке по первому символу имени параметра делается выбор формата. Целый формат выбирается символами I, J, K, L, M и N, текстовый — символом T, остальные символы указывают на вещественный формат. Имеется аналогичная программа, осуществляющая вывод по имени набора NAME

```
call SC1323(NAME)
```

Если имя отсутсвует в списке имен, то распечатываются все наборы объектов. Можно распечатывать не все параметры объектов, а только L1 первых, для этой цели служит подпрограмма

```
call SC1322(ID,L1)
```

Кроме того, набор объектов ID можно распечатать по строкам

```
call SC1315(ID)
```

Распечатка отношения ID осуществляется подпрограммой

```
call SC1313(ID)
```

Если ID отсутствует в списке идентификаторов отношений, то выводятся все отношения. При выводе предполагается, что все параметры отношения вещественны.

Следующие три подпрограммы практически не бывают нужны, но приводятся в описании для полноты. Состояние структуры данных СОСНы выводится с помощью подпрограммы

```
call SC1321
```

Состояние структуры данных СОСНы на данный момент можно сохранить в файл с полным именем NAMEF с помощью подпрограммы

```
call SC1330(NAMEF)
```

Состояние структуры данных СОСНы можно восстановить из файла с полным именем NAMEF с помощью подпрограммы

```
call SC1331(NAMEF)
```

Дополнительные сообщения подпрограмм СОСНы о неверно заданных параметрах подпрограмм сопровождаются вызовом подпрограммы SC1332, которая обеспечивает вызов системных подпрограмм VAX VMS

для вывода цепочки имен подпрограмм, из которых осуществляется неверный вызов. Подпрограмма SC1332 должна быть заменена при переходе к другой операционной системе. Минимальная замена — это использование "пустой" подпрограммы, хотя опыт показывает, что отладка программы реконструкции заметно ускоряется, когда сразу становится известно место в программах пользователя, из которого следует некорректный вызов. Печать из программы SC1332, как и другие сигнальные сообщения СОСНы, снабжена счетчиком числа печатей. Если количество сообщений подпрограммы превышает число MPRINT из общей области /SND326/, вывод сообщений прекращается.

4.7 Формат записи событий в файл

Экспериментальные данные, зарегистрированные детектором, должны быть некоторым образом упакованы и сохранены на магнитных накопителях (диски, ленты, видиоленты). При обработке данные о событиях последовательно считывается программой обработки, по ним проводится реконструкция, результаты обработки упаковываются и записываются во вновь создаваемый файл для сохранения с целью последующей обработки. Пакет программ СОСНА, обеспечивающий процесс обработки события может быть абсолютно не связан с форматами файла с событием и способом упаковки параметров события. Однако формат файла с событием и несколько обслуживающих подпрограмм разработанных для обработки данных детектора СНД [6] могут быть полезны для пользователей, работающих в условиях с ограниченной дисковой памятью и ограниченной памятью на магнитных накопителях, поэтому описание группы таких подпрограмм приводится в данной работе.

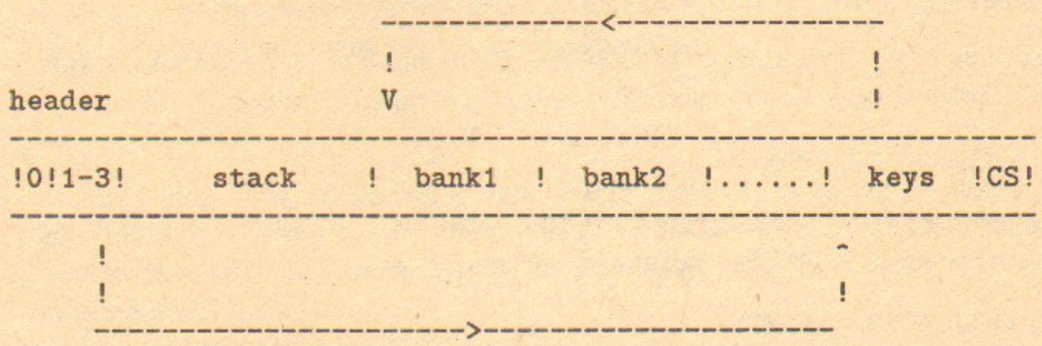

Рис. 3. Структура посылки с событием детектора СНД. Стрелками показан метод доступа к информационным банкам.

Основное соображение, которое использовалось при этой разработке, состояло в том, что все величины должны быть упакованы в целые числа длиной 2 байта, поскольку в эксперименте СНД за редким исключением нет параметров, для измерения которых использовалось бы большее количество байт. В этом параграфе все переменные, имена которых начинаются с символа U, будут описаны как INTEGER*2. Все адреса — это индексы в массиве ULET из общей области /SND300/, в которой событие упаковывается. Этот массив структурно делится на следующие части (рис. 3): шапку, стек, информационые банки, ключи, контрольную сумму. Шапка — это 4 целых числа, функциональное назначение которых независимо от типа записи следующее:

```
UNLET   - адрес контрольной суммы (длина события)
ULET(1) - тип.*256 + год*16 + месяц
ULET(2) - число*256 + номер захода
ULET(3) - адрес ключей.
```

Здесь "тип" — целое число, которое характеризует тип записи; "год" — последняя цифра года; "месяц" — номер месяца; "число" — день месяца; "номер захода" — это номер экспериментального захода в данный день. Стек — это банк фиксированной длины для данного типа записи, начинающийся с ULET(4). Доступ к информации, записанной в стек, наиболее быстрый. Информацбионные банки — это записи переменной длины, их структура не зависит от типа посылки. Все они устроены одинаково: первое число содержит длину банка, включая это число, затем идет информация, упакованная в соответствии с идентификатором ключа к банку. Длина банка переменна, она зависит от события.

Область ключей находится после области информационных банков. В буфере ключей первый элемент — количество ключей в данном событии, а затем располагается информация о каждом ключе, которая состоит из двух чисел:

— идентификатор банка;

— адрес начала банка.

Таким образом, количество ключей равно количеству информационных банков, порядок следования ключей и их банков произвольный. Буфер с событием замыкает контрольная сумма, которая может использоваться для контроля за сохранностью информации.

Перед началом работы с записями пользователь должен инициализировать общие области /SND300/, /SND301/, /SND303/. Программа пользователя должна обеспечить открытие файла на магнитном накопителе и считывание записи с очередным событием в общую область

/SND300/. Для того, чтобы получить доступ к информационному банку с идентификатором ID (INTEGER*4), можно воспользоваться вызовом подпрограммы

```
call SR1300(ID,UA*,CODE*)
```

UA — (INTEGER*2) адрес начала банка; CODE — (INTEGER*4) код завершения, который может принимать следующие значения:

1 — банк отсутствует в событии,

2 — идентификатор отсутствует в списке /SND301/.

Пользователь может записать в буфер с событием банк с идентификатором ID (INTEGER*4)

```
call SW1300(ID,UB,CODE*)
```

UB — (INTEGER*2) адрес начала буфера пользователя с подготовленным банком (UB(1) — длина банка); CODE — (INTEGER*4) код завершения, который может принимать следующие значения:

1 — банк с таким идентификатором существовал, он заменен;

2— событие превысило разрешенную длину буфера, запись не производится;

3 — идентификатор отсутствует в списке /SND301/.

В конце обработки события пользователь может уплотнить посылку, используя список идентификаторов,

```
call SW1302(N,UM,CODE*)
```

N — (INTEGER*4) количество банков, которые нужно сохранить; UM(N) — (INTEGER*2) массив идентификаторов банков, которые нужно сохранить; CODE — (INTEGER*4) код завершения, который может принимать следующие значения:

1 — банк отсутствует,

2 — идентификатор отсутствует в списке /SND301/.

Форматный вывод буфера с событием по ФОРТРАН — каналу LUN (см. /SND700/) осуществляет программа

```
call SP1300(M)
```

M — (INTEGER*4) флаг подробности печати (если M=1, то подробная печать, включая информационные записи, иначе печать без информационных записей).

5 Общие области

Общие области пакета программ СОСНА делятся на две группы. К первой относятся общие области, в которых поддерживается структура данных СОСНы. Все эти области находятся в файле SCOM00.FOR и могут включаться пользователем в свою программу с помощью INCLUDE. На VAX кластере ИЯФ СО РАН этот файл находится в директории

```
disk$d5:[ivanchenko.snd_lib]
```

В данном описании все переменные в этих общих областях по умолчанию имеют длину 4 байта, все адреса являются индексами массивов переменных длиной 4 байта. Все длины массивов могут быть изменены пользователем (см. п.4) или автором в новой версии программы.

/SND320/NO,NDESK(6,200) — справки о наборах объектов,

NO—количество наборов объектов.

справка о каждом наборе содержит 6 чисел:

1,2—(CHARACTER*8) имя набора;

3—количество параметров объекта;

4—количество объектов набора в событии;

5—указатель (адрес-1) на начало банка с параметрами набора в буфере общей области /SND321/;

6—указатель (адрес-1) на начало банка с именами параметров объектов набора в буфере общей области /SND322/.

/SND321/LOB,NOB,BUFOB(40000) — буфер с банками, содержащими наборы объектов СОСНы,

LOB—длина заполнения буфера;

NOB—номер последнего набора, который заносился в буфер;

/SND322/LNA,BNAME(10000) — буфер с банками, содержащими имена параметров объектов.

LNA—длина заполнения буфера.

/SND323/NREL,IREL(8,200) — справки об отношениях между наборами,

NREL—количество отношений.

Справка о каждом отношении содержит:

1,2—(CHARACTER*8) имя отношения;

3—идентификатор 1-го набора в отношении;

4—идентификатор 2-го набора в отношении;

5—признак определения отношения (если 0 — нет, иначе да);

6—указатель (адрес-1) на начало начала массива ключей отношения в буфере общей области /SND324/;

7—число параметров отношения (может быть и 0);

8—указатель (адрес-1) на начало банка с именами параметров отношений в буфере общей области /SND322/.

/SND324/LREL,NRB,IADR(2,2000) — буфер, содержащий ключи к банку данных с отношениями.

LREL—длина заполнения буфера;

NRB—номер последнего отношения, записанного в банк;

IADR—ключи к банкам данных в общей области /SND325/:

1—число объектов набора 1, имеющих отношение к данному объекту набора 2;

2—указатель (адрес-1) на начало списка объектов набора 1, имеющих отношение к данному объекту набора 2 в буфере общей области /SND325/, сразу после этого списка расположен банк с параметрами этого отношения.

/SND325/LRL,IBUFR(40000) — буфер с банками данных по отношениям.

LRL—длина заполнения буфера.

/SND326/MNO,MLOB,MNA,MNR,MLREL,MLRL,MPRINT — максимальные длины буферов и ограничение на число печатей.

В данной версии пакета значения этих переменных следующие:

MNO=200 — /SND320/;

MLOB=40000 — /SND321/;

MNA=10000 — /SND322/;

MNR=200 — /SND323/;

MLREL=2000 — /SND324/;

MLRL=40000 — /SND325/;

MPRINT=100.

/SND700/LUN — номер канала для вывода сообщений и распечаток, по умолчанию равен 6.

Другая группа общих областей предназначена для обмена с магнитными накопителями. Она не используется в основных подпрограммах пакета COCHA. В той же директории, что и SCOM00.FOR, имеется файл SND300.FOR, в котором определены эти общие области для пользователей детектора СНД. Пользователи других групп сами должны инициализировать эти общие области, определить длины буферов, считывать очередное событие в буфер. Все переменные в этих общих областях по умолчанию описаны как INTEGER*2.

/SND300/UNLET,ULET(WLET) буфер с событием.

/SND301/WCAY,WLET,WLETT,QCAY(WCAY) — контрольная информация,

WCAY — максимальное число идентификаторов банков;

WLET — максимальная длина буфера посылки;

WLETT — максимальное число типов посылок;

QCAY — (CHARACTER*8) имена банков.

/SND303/UCAYL,UCA(2,WCAY) временный буфер ключей.

6 Список подпрограмм

Ниже приводится полный список подпрограмм пакета COCHA без описания, которое приведено в пп.4.—4.6. Все выходные параметры подпрограмм будут помечены символом *. По умолчанию предполагается ФОРТРАН определение переменных (INTEGER*4 и REAL*4), текстовые переменные начинаются с символов NAM, переменные INTEGER*2 — с символа U.

SC1300(NAME,L,NAMP,ID*) — регистрация набора.

SC1301(NAME,ID*,L*) — доступ к банку данных с параметрами набора.

SC1302(ID,A) — запись объекта в банк.

SC1303(ID,LM,A*,K*) — считывание набора в буфер пользователя.

SC1304(NAME,ID1,ID2,L,A,ID*) - регистрация отношения.

SC1305(NAME,ID*,L*) — доступ к отношению.

SC1306(ID,K2,L,IB,K2M,IBP) — запись отношения в банк.

SC1307(ID,K2,L*,IA*,IAP*,*M) — доступ к отношению.

SC1308(ID12,ID21) — обращение отношения.

SC1309(ID12,ID23,ID13) — произведение отношений.

SC1310(ID1,ID2,ID) — сложение отношений.

SC1311(ID) — инициализация.

SC1312(ID) — распечатка объектов набора по столбцам.

SC1313(ID) — распечатка отношения.

SC1314(NAME,L) — переопределение максимальных длин буферов.

SC1315(ID) — распечатка объектов набора по строкам.

SC1316(ID,K*,IA*) — доступ к банку с объектами набора.

SC1317(ID,K2,LM,IA,LMP,IAP,L,*M) — считывание отношения в буфер.

SC1318(ID,K2,A) — запись нового списка параметров отношения в банк.

SC1319(USER,N,N1,N2,N3,N4,N5,N6) — вызов программы пользователя USER.

SC1320(L,IA*) — запрос участка буфера общей области /SND322/.

SC1321 — распечатка статуса структуры данных.

SC1322(ID,L1) — распечатка объектов по столбцам.
SC1323(NAME) — распечатка объектов по столбцам.
SC1324(ID,K,IA*) — резервирование банка под объекты набора.
SC1325(ID,K,A) — запись объектов в набор.
SC1326(ID,K) — удаление объекта из набора.
SC1327(ID,K2,L1) — сокращение списка дочерних объектов отношения.
SC1328(ID,K,A) — модификация объекта.
SC1329(ID,K,NM,IDM,ND,IDD) — удаление объекта из набора.
SC1330(NAMEF) — сохранение структуры данных в файл.
SC1331(NAMEF) — восстановление структуры данных из файла.
SC1332 — распечатка цепочки вызовов подпрограмм.
SCOM00 — общие области пакета программ СОСНА.
SR1300(ID,UA*,CODE*) — доступ к информационному банку.
SW1300(ID,UB,CODE*) — сохранение информационного банка.
SW1302(N,UM,CODE*) — уплотнение события.
SP1300(M) — распечатка события.
SND300 — общие области для ввода/вывода событий детектора СНД.

7 Использование пакета программ СОСНА для реконструкции событий детектора СНД

На основе пакета программ СОСНА создана программа реконструкции событий детектора СНД [6] и пакет программ дальнейшей обработки этих событий. Поскольку сам пакет программ СОСНА совершенствовался в период этой разработки, то удалось опробовать и сравнить различные варианты его использования. Основной вывод состоит в том, что в операционной системе VAX VMS наиболее эффективный способ работы с этой структурой данных состоит в использовании подпрограммы SC1319, предоставляющий пользователю доступ непосредственно к банкам с наборами объектов и не приводящей при этом к заметным накладным расходам. Оказалось удобным также каждому пользователю проводить регистрацию своих объектов и отношений в одной определенной подпрограмме. Размещение констант, используемых при реконструкции, в неизменяемых объектах СОСНы (см. п. 4.1) принесло также определенную пользу.

При разработке программы реконструкции важную роль играли диаграммы объектов и отношений, использование которых позволило разработать проект до начала его реализации, а затем, по ходу работы, достаточно безболезненно вносить в него оперативные изменения. В частности, удалось из программы реконструкции событий выделить некоторую часть, которая обеспечивает предварительный поиск частиц и их параметров. Эта часть программы используется в эксперименте в реальном времени и подготавливает информацию для программного отбора событий, который принято называть “Третичным триггером” эксперимента. Скорости обработки событий моделирования для детектора СНД приведены в Табл. 1. Несколько упрощенный вид диаграммы объектов и отношений для этой части программы приведен на рис. 4.

Таблица 1.

Скорость полной реконструкции событий моделирования для детектора СНД (REC) и скорость обработки on-line программой реконструкции (TR3). Моделирование выполнено для энергии пучков 510 МэВ, расчеты производились с использованием VAXserver-3300.

Процесс	e^+e^-	$\pi^+\pi^-$	$\pi^0\gamma$	K_SK_L	K^+K^-
TR3 (Гц)	46	30	45	25	20
REC (Гц)	17	11	18	9	4

Ниже в качестве примера приводится часть программы on-line реконструкции событий. В этой части, в соответствии с диаграммой на рис. 4, определяется список кластеров в калориметре, а затем находятся частицы. Основной вызов всех подпрограмм осуществляется с помощью подпрограммы SC1319, приведены отрывки из текстов двух таких подпрограмм. Отметим, что каждая подпрограмма содержит в своем начале заметную область описаний, которая на первый взгляд может показаться громоздкой, но по существу является стандартной и может быть сравнительно легко подготовлена пользователем по образцам.

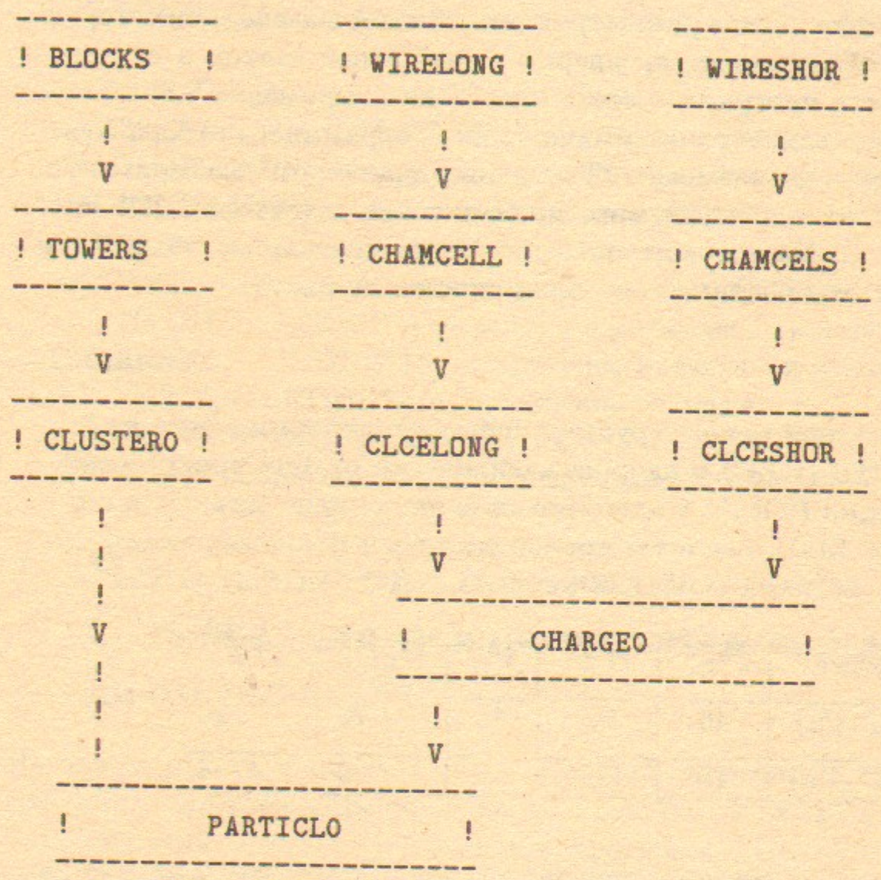

Рис. 4. Диаграмма объектов и отношений, реализованная в программе быстрой реконструкции событий детектора СНД. В прямоугольниках показаны имена наборов объектов, стрелки — отношения между наборами.


```
C=================================================
C       MAIN ROUTINE  OF 3-D LEVEL TRIGGER
C=================================================

      SUBROUTINE Rec_Trigger3

C=================================================
C       USER ROUTINES
C=================================================

      EXTERNAL CL3302, CL3303, CL3307, CL3312

C=================================================
C       Including COCHA COMMONs
C=================================================

      INCLUDE 'SCOMOO.FOR'

C=================================================
C       LIST OF NAMES OF PARAMETERS OF ENTITIES
C=================================================

      PARAMETER (NPC=16, NPCH=11, NPP=21)

      CHARACTER*4 CLUST(NPC)
     1/'FICL','@TCL','SGFI','SGTE','ECT ','EC1 ','EC2 ','EC3 ',
     2 'ECO ','ECO1','ECO2','ECO3','XIE2','N1  ','NOUT','NPAR'/
      CHARACTER*4 CHARGE(NPCH)
     1/'NLDC','NSC1','NSC2','KSDC','ITCO','NPAR','NPR1','NPR2',
     2 'IOU1','PHR1','PHL1'/
      CHARACTER*4 PART(NPP)
     1/'NCLO','NCHO','NCH1','NCO1','IPAR','EPAR','PHIP','@ETP',
     2 'ZO  ','NP1 ','IPH1','ITE1','IPH2','ITE2','E1F ','E2F ',
     3 'E1T ','E2T ','EDUB','XI2G','NCL '/

        ..........................

C=================================================
C       LIST OF NAMES OF PARAMETERS OF RELATIONSHIPS
C=================================================
```

```
      PARAMETER (LTCL=2)
      CHARACTER*4 BLOC2(LTCL)/'DELT','DECL'/

C==================================================
C       FIRST ENTRY
C==================================================

      LOGICAL KLU/.TRUE./

      IF(KLU)THEN
        KLU=.FALSE.

C==================================================
C    OPEN EXISTING SETS OF ENTITY
C==================================================

        CALL SC1301('BLOCKS  ',NBL,LPB)
        CALL SC1301('TOWERS  ',NTO,LPT)

        ..................

C==================================================
C    DEFINE NEW SETS OF ENTITY
C==================================================

        CALL SC1300('CLUSTERO',NPC,CLUST,NCL)
        CALL SC1300('CHAMCELL',NPS,SELL,NSL)
        CALL SC1300('CHAMCELS',NPS,SELL,NSS)
        CALL SC1300('CHARGEO ',NPCH,CHARGE,NCH)
        CALL SC1300('PARTICLO',NPP,PART,NPA)

        ..................

C==================================================
C    DEFINE NEW RELASHIONSHIPS
C==================================================

        CALL SC1304('CLOTOWER',NCL,NTO,LTCL  ,BLOC2,NCLTO)
        CALL SC1304('CLOPARTO',NCL,NPA, 0    ,BLOC2,NCOTO)
```



```
        CALL SC1304('TOWERCLO',NTO,NCL,LTCL  ,BLOC2,NTOCL)

        .................

      END IF

C============> ETOT, PRT, PLR OF CALORIMETER

      CALL SC1319(CL3302,1,NTO)

C============> SET OF ENTITY "CLUSTER1"

      CALL SC1319(CL3303,2,NTO,NCL)

C============> FIND OUT PARAMETERS OF CLUSTER1

      CALL SC1308(NTOCL,NCLTO)
      CALL SC1319(CL3307,2,NTO,NCL)

      ........................

C===========> SET OF ENTITY "PARTICLO"

      CALL SC1319(CL3312,5,NCL,NCH,NSL,NSS,NPA)
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE CL3302(B,IB,L,K)

C============================================================
C        DETERMINATION ETOT, PTRX, PTRY, PTRZ
C          USING ENERGY DEPOSITION IN TOWERS
C============================================================

      IMPLICIT INTEGER *2 (U-W)
      COMMON/SND300/UNLET,ULET(13)
      COMMON/SND305/UFLR,UFLW,UNR,UNW,UTYPR,UTYPW,UTLETT(2,2)
      COMMON/SND330/FIGR(3,60),TEGR(5,16)
      DATA EMAX/32000./
```

```
C==========> Bank with TOWERS parameters

      DIMENSION B(L,1),IB(L,1)

      ETOT=0.
      PTRX=0.
      PTRY=0.
      PTRZ=0.
        DO1 I=1,K
          E=B(3,I)
          IF=IB(1,I)
          IT=IB(2,I)
          ETOT=ETOT+E
          PTRX=PTRX+E*TEGR(3,IT)*FIGR(3,IF)
          PTRY=PTRY+E*TEGR(3,IT)*FIGR(2,IF)
          PTRZ=PTRZ+E*TEGR(4,IT)
1       CONTINUE
      PTR=SQRT(PTRX**2+PTRY**2)*10.
      ETOT=ETOT*10.
      PTRZ=PTRZ*10.
      IF(ETOT.GT.EMAX)ETOT=EMAX
      IF(PTR .GT.EMAX)PTR =EMAX
      IF(PTRZ.GT.EMAX)PTRZ=EMAX
      IF(PTRZ.LT.-EMAX)PTRZ=-EMAX
      ULET(11)=INT(ETOT)
      ULET(12)=INT(PTR)
      ULET(13)=INT(PTRZ)
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE CL3303(BT,IBT,LT,KT,BC,IBC,LC,ICL)

C=================================================================
C    FIND OUT CLUSTERO IN THE CALORIMETER SND
C      ICL - NUMBER OF CLUSTERS
C=================================================================

C==========> Bank with TOWERS parameters
```



```
      DIMENSION BT(LT,1),IBT(LT,1)

C==========> Bank with CLUSTERO parameters

      DIMENSION BC(LC,1),IBC(LC,1)

C================================================
C.......FIRST ENTRY
C================================================

      LOGICAL KLU/.TRUE./

      IF(KLU)THEN

C==========> OPEN RELASHIONSHIP

        CALL SC1305('TOWERCLO',NTOCL,LTOCL)

C==========> DEFINE CONSTANTS

        CALL SC1301('PARAMCLO',NPO,NPARO)
        IAAP=NDESK(5,NPO)
        GRAD=180./3.14159265
        ETH1=BUFOB(IAAP+1)
        DTET=BUFOB(IAAP+3)
        ECH=BUFOB(IAAP+5)
        KLU=.FALSE.
      END IF

        ..................

C================================================
C.......CICLE THROUW TOWERS
C================================================

20    CONTINUE

      ..................
```

```
      DO 21 I=II1,II2
        IF(IBT(7,I).GE.1.AND.IBT(14,I).EQ.0)THEN
          IF(BT(3,I).GT.ET)THEN
            JT=I
            ET=BT(3,I)
          END IF
        END IF
21    CONTINUE
      IF(JT.EQ.0)RETURN

C================================================
C        THERE IS FREE TOWER NUMBER JT
C================================================

      IRT=1

      ...................

C================================================
C.......CICLE THROUW TOWERS
C================================================

      DO 24 J=1,KT
        IF(J.NE.JT)THEN

        ......................

24    CONTINUE

C=====================================================
C           THERE IS GOOD NEUTRAL CLUSTER
C          BOOKING ENTITY AND RELASHIONSHIP
C=====================================================

      IF(ETOT.LT.ECH)THEN
        ICL=ICL+1
        DO 25 I=1,IRT
          J=IRET(I)
          if(IBT(7,J).lt.4.or.BRET(1,I).eq.0.)IBT(14,J)=ICL
          IF(IBT(12,J).LE.0)IBT(12,J)=ICL
```



```
25        IBT(11,J)=IBT(11,J)+1
        DO 9 I=1,LC
9         BC(I,ICL)=0.
        BC(5,ICL)=ETOT

C============> Save relashionship

        CALL SC1306(NTOCL,ICL,IRT,IRET,KT,BRET)
      END IF
      GO TO20
      END
```

В заключение автор хотел бы выразить большую благодарность А.Д. Букину и Е.В. Пахтусовой за многократные полезные обсуждения.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (код проекта 93-02-3295).

8 Литература

1. M.G.Green, The ADAMO Data System, CERN/DD/US/131 (1989).

2. V.Blobel, The BOS System. Dynamic Memory Management, DESY Internal Report N R1-88-01 (1988).

3. A.S.Johnson et al., JAZELLE: An Enhanced Data Management System for High Energy Physics, SLAC-PUB-5231 (1990).

4. S.Youssef , in: Proc. Workshop on Detector and Event Simulation in High Energy Physics (Amsterdam, 1991), NIKHEF-H, Amsterdam, 1991, p.172.

5. R.Brun ans J.Zoll, ZEBRA — Data Structure Management System, CERN Program Library Q100 (1989).

6. V.M.Aulchenko et al., Preprint INP 87-36 (Novosibirsk, 1987);
   L.M.Barkov et al., in: Proc. 14th Intern. Conf. on High Energy Particle Accelerators (Tsucuba, 1990);
   L.M.Barkov et al., in: Proc. 1991 IEEE Part. Acc. Conf. (San Francisco, 1991);
   S.Seredyakov et al., in: Proc. 5th Intern. Conf. on Instrumentation for Colliding Beam Physics (Novosibirsk, 1990), World Scintific, Singapore, 1990, p.360;
   S.Serednyakov et al., in: Proc. Workshop on Physics and Detectors for DAΦNE (Frascati, 1991), Frascati, 1991, p.605.

7. P.P.Chen, "The Entity-Relashionship Model Toward a Unified View of Data", ACM Trans. on Database Systems 1 (1976).

В.Н. Иванченко

СОСНА—пакет программ для структурирования экспериментальных данных

Ответственный за выпуск С.Г. Попов
Работа поступила 1 марта 1994 г.

Сдано в набор 1 марта 1994 г.
Подписано в печать 21 марта 1994 г.
Формат бумаги 60×90 1/16 Объем 1.7 печ.л., 1.4 уч.-изд.л.
Тираж 200 экз. Бесплатно. Заказ N° 25

Обработано на IBM PC и отпечатано на
ротапринте ИЯФ им. Г.И. Будкера СО РАН,
Новосибирск, 630090, пр. академика Лаврентьева, 11.