ОРДЕНА ЛЕНИНА
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М. В. Келдыша
Российской академии наук

Т. П. Баранова, В. Ю. Вершубский

Анализатор программ, написанных на языке ФОРТРАН

Москва
2005

Т.П.Баранова, В.Ю.Вершубский

Анализатор программ, написанных на языке ФОРТРАН♦

Аннотация

В данной работе представлена интерактивная программа - анализатор, предназначенная для получения сведений о программах, написанных на языке ФОРТРАН 77. Анализатор предоставляет пользователю по его запросам такую информацию, как перечень описанных в программе использованных и неиспользованных программных единиц, перечень объявленных в программной единице, использованных и неиспользованных переменных, сведения о типах переменных и способах задания типа, сведения о циклах, о метках, о соответствии фактических параметров процедур и функций их формальным параметрам, и т. п. Работа анализатора базируется на использовании библиотеки классов Си++ пакета SAGE. Собранные с помощью анализатора сведения об исследуемой ФОРТРАН-программе могут быть использованы как для ее модификации и оптимизации, так и для распараллеливания. Анализатор работает в среде MS Windows 2000.

T.P.Baranova, V.Yu.Vershoubskii

A Program to analyze texts written in the FORTRAN language

Abstract

An interactive program – analyzer is described designed todeliver to a user on his request information on a large program written in the FORTRAN 77 language. The analyzer delivers to the user such information as a list of described in the project used and unused program units, lists of declared in the program unit used and unused variables, information on variable kind and type and how the type is set. The analyzer can deliver data on cycles, labels, on agreement of parameters in calls to procedures (functions) and correspondent dummies in descriptions, etc. The information on a FORTRAN-program gained using the analyzer could be used for modification and optimization of the program under investigation, or for its parallelization as well. The analyzer based on usage of the C++ class library of the SAGE packet and operates in the MS Windows 2000 media.

♦ Работа выполнена при частичной финансовой поддержке программы российских научных школ РИ-112/001/278.

Содержание

1. Введение 3
2. Подготовка входных данных 5
3. Работа с анализатором 5
3.1 Общая структура окон 5
3.2 Пункты меню “Запросы по проекту” 13
3.2.1 Список COMMON-блоков 13
3.2.2 Лишние метки в процедурах 14
3.2.3 Неиспользуемые переменные 15
3.2.4 Размещение процедур по файлам 15
3.2.5 N-мерные массивы 16
3.2.6 Тесногнездовые циклы 16
3.2.7 Массивы максимальной размерности 17
3.2.8 Процедуры, имеющие READ 17
3.3 Пункты меню “Запросы к процедурам” 17
3.3.1 Вызываемые процедуры 17
3.3.2 Список COMMON блоков 18
3.3.3 Список обращений 18
3.3.4 Список меток 18
3.3.5 Лишние метки 19
3.3.6 N-мерные массивы 19
3.3.7 Неиспользуемые переменные 19
3.4 Контекстные меню 19
4. Состав анализатора и вспомогательные программы. 24
Заключение 25
Литература 25
Приложение 1. Пример INI-файла 26
Приложение 2. Пример графа вызовов для процедуры ERRO 27

1. Введение

Развитие вычислительной техники за последнее десятилетие привело к появлению довольно большого числа многопроцессорных вычислительных установок. Доступность этих установок для пользователей, в свою очередь, породила интерес к преобразованию накопленных за прошлые годы больших программ, созданных для последовательных машин, в эффективные “параллельные” программы, пригодные для работы на многопроцессорных системах. Как правило, такое распараллеливание пока производится “вручную”, и в случае успеха приводит к значительному увеличению скорости решения вычислительных задач.

Работы по созданию систем, автоматизирующих в той или иной степени преобразование последовательных программ в параллельные, ведутся давно, и в настоящее время интенсивность таких исследований не снижается (Vray [1], Parawise [2], CAPTools [3], POLARIS [4], SUIF [5]). Это положение связано, по-видимому, с тем обстоятельством, что решить проблему автоматического распараллеливания программ в приемлемых, с практической точки зрения постановках, пока не удается. В плане этих работ средства анализа, помогающие человеку разобраться в структуре программы, и облегчающие ее преобразование, могут оказаться весьма полезными. Тем более что часто преобразование последовательных программ в параллельные осуществляют не те, кто первоначально эти программы создавал. По мере накопления опыта работы с такими средствами глубина осуществляемого ими анализа может возрастать.

В данном препринте описываются возможности первого варианта интерактивной программы-анализатора, осуществляющей исследование статических свойств программ, написанных на языке ФОРТРАН 77. Анализатор работает в операционной среде Windows 2000. Он способен справляться с большими проектами, имеющими размер порядка десятков тысяч предложений исходного текста. Анализатор предоставляет пользователю сведения, об описанных в проекте программных единицах (использованных и неиспользованных), об объявленных в программных единицах переменных (использованных и неиспользованных), о разновидностях и типах переменных, о способе задания типа переменных, о взаимных обращениях программных единиц. Анализатор предоставляет сведения о циклах и о переменных с индексами, управляемыми этими циклами, об индексных выражениях, и т.п.

Работа анализатора базируется на использовании пакета программ SAGE. В работе [6] мы описали опыт нашего ознакомления с этим пакетом и некоторые соображения о возможности его использовании для достижения поставленных целей. В данной работе использована версия SAGE 1.9 от 05.1995 [7].

Ниже раскрываются возможности анализатора через описание интерфейса взаимодействия пользователя с программой.

2. Подготовка входных данных

Для того чтобы можно было воспользоваться анализатором, надо произвести над исследуемой программой определенную подготовительную работу.

Если в исследуемой программе используются предложения include, то надо сначала получить единый файл программы, осуществив слияние всех используемых файлов. Для этого имеется отдельная утилита.

Если размер полученной в результате слияния программы превосходит 4500 предложений, то надо полученный в результате текст разбить на фрагменты, не превышающие указанный выше размер. При этом программная единица (SUBROUTINE, FUNCTION) не должна начинаться в одном фрагменте и продолжаться в другом. (Ограничение на размер фрагмента диктуется парсером, входящим в состав SAGE-пакета.)

Каждый из полученных фрагментов текста необходимо пропустить через парсер – программу пакета SAGE, который переводит исходный текст в промежуточное машинно-независимое представление, имеющее вид совокупности древовидных графов. Выходные файлы парсера должны иметь расширения .dep.

Исследуемую программу надо представить в виде проекта, состоящего из одного или нескольких dep-файлов. Для этого в подходящем редакторе создается текстовый файл, содержащий перечень имен dep-файлов, входящих в состав проекта. Имя файла проекта должно иметь расширение .prj. При запуске анализатора потребуется указать имя этого файла, описывающего исследуемый проект.

3. Работа с анализатором

3.1 Общая структура окон

После запуска анализатор раскрывает начальное окно, приведенное на Рис. 1

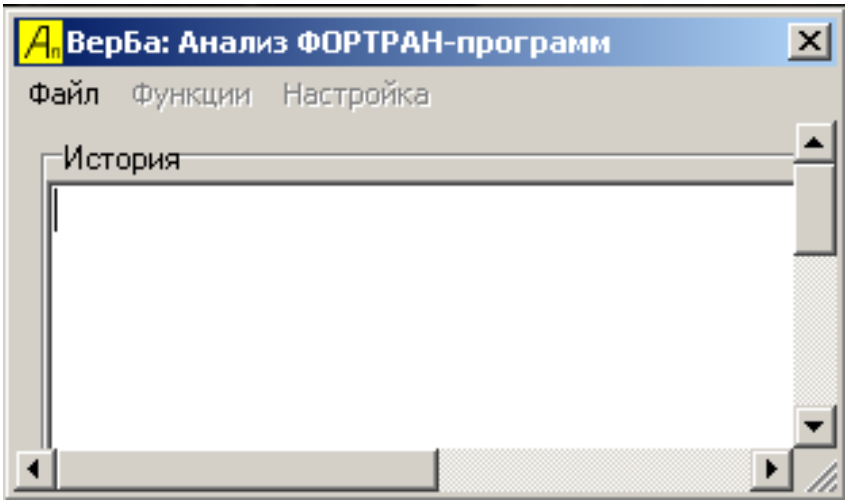

Рис. 1

В этом окне в верхней строке располагается главное меню, состоящее из 3-х функций: **`Файл, Функции, Настройка`**

В исходном состоянии доступен только один пункт главного меню - **Файл,** а в соответствующем подменю, только пункт - **Открыть проект**. После выбора команды **Файл->Открыть проект** открывается окно типового графического диалога (Рис. 2), позволяющего перемещаться по дереву файловой системы для выбора требуемого файла проекта. По умолчанию в окне диалога высвечиваются только имена файлов с расширением .prj.

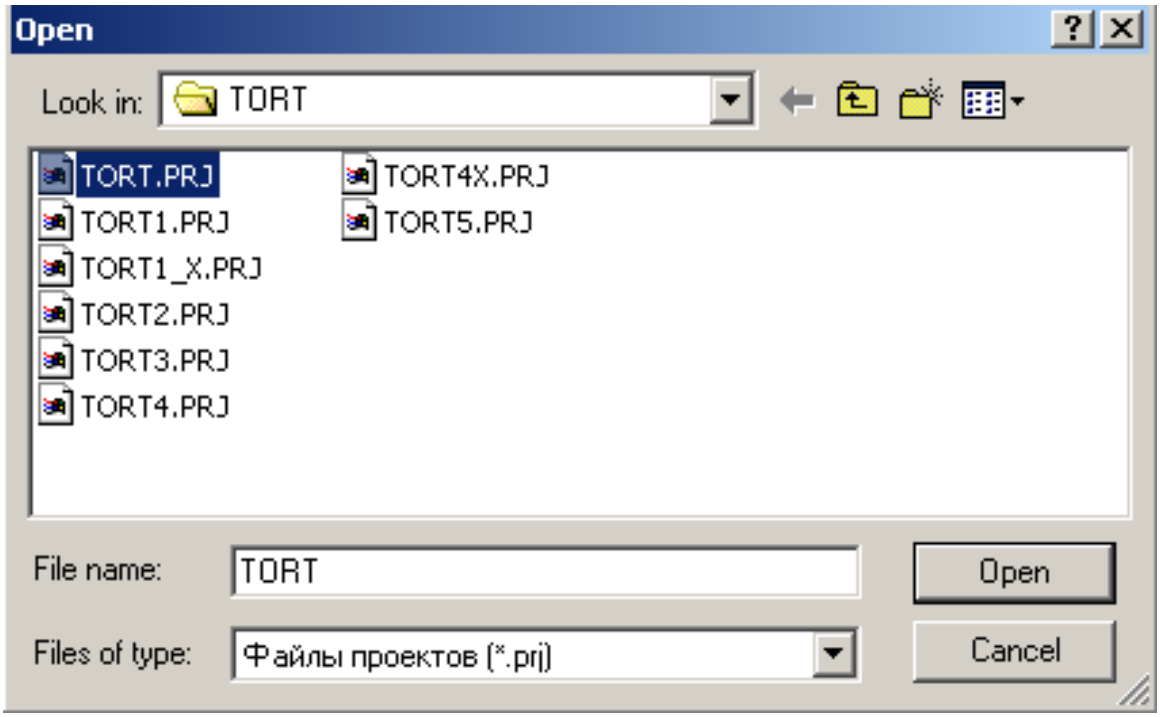

Рис. 2

После открытия проекта в окне "**История**" печатается сообщение:

"-------- Проект <имя>.prj открыт --------"

Анализатор может работать только с одним открытым проектом, поэтому после открытия соответствующего файла функция **Файл->Открыть проект** становится недоступной, а доступной становится функция **Файл->Закрыть проект**. Для того чтобы перейти к другому проекту необходимо закрыть текущий проект.

После открытия проекта также становится доступной функция меню **Функции->Собрать сведения по файлам проекта** (см. Рис. 3).

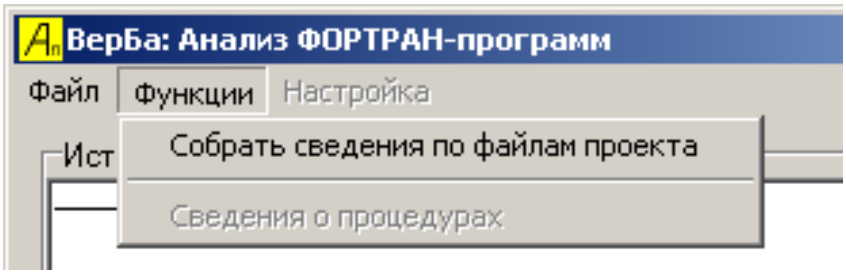

Рис. 3

Сбор сведений по проекту включает в себя получение графа потока управления, информации о программных единицах, информации о переменных в программных единицах, сведения о циклах, и т.п., и может занимать значительное время. По мере обработки файлов проекта в окно **"История"** выдается имя соответствующего файла. После сбора всех сведений в окно **"История"** выдается сообщение: **"Сведения по проекту собраны"** (см. Рис. 4).

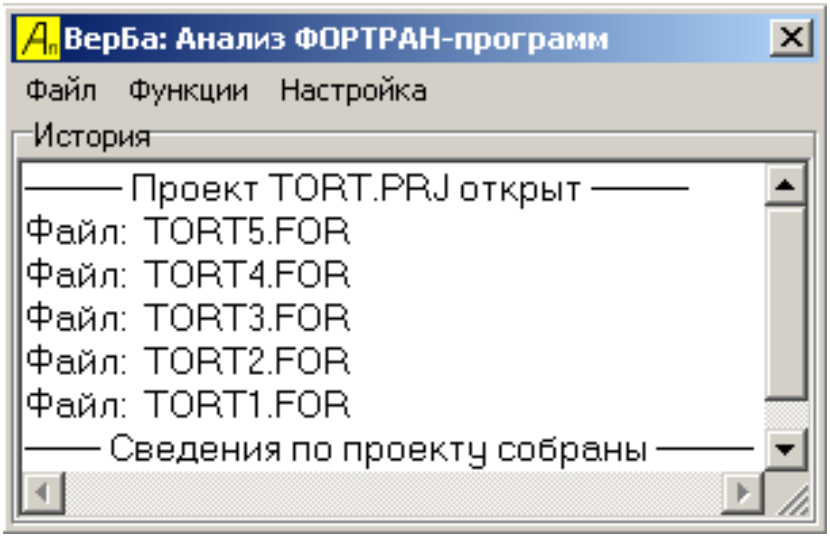

Рис. 4

После этого становится доступной третья функция меню:

Настройка->Цепочка вызовов процедур

При использовании этой функции раскрывается вспомогательное окно, позволяющее задать условия сбора цепочки вызовов процедур, а именно:

- исключить из рассмотрения библиотечные функции, встроенные в язык. По умолчанию – исключить.
- исключить из рассмотрения MPI-функции. По умолчанию – исключить.
- исключить из рассмотрения функции, указанные пользователем. (По умолчанию все программные единицы, описанные в проекте, включаются в рассмотрение при построении цепочки вызовов.)

Пример окна настройки приведен на Рис.5.

Условия сохраняются в специальном INI-файле при закрытии вспомогательного окна. Если при открытии проекта INI-файл отсутствует, то он будет создан, и в него будут занесены условия, используемые по умолчанию. Если INI-файл существовал, то будут соблюдены заданные в нем условия. INI-файл представляет собою обычный текстовый файл, который можно редактировать вне программы текстовым редактором. Структура INI-файла приведена в Приложении 1.

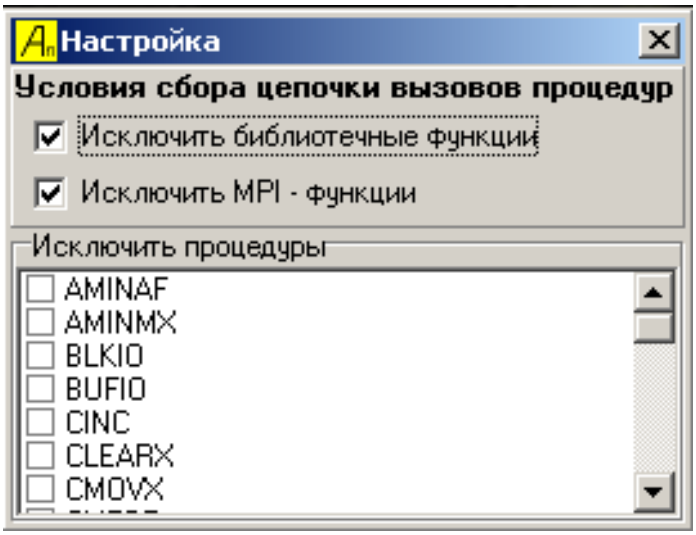

Рис. 5

После сбора сведений функция "**Собрать сведения по файлам проекта**" становится недоступной, а открывается доступ к функции "**Сведения о процедурах**", (см. Рис. 6).

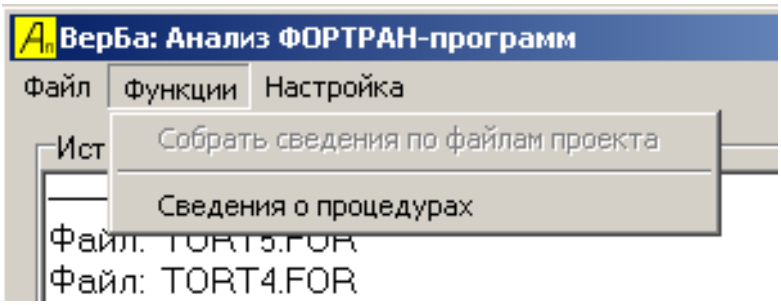

Рис. 6

При выполнении этой функции раскрывается окно с заголовком "**Сведения по процедурам**", в котором будут отображаться результаты всех последующих запросов (см. Рис. 7).

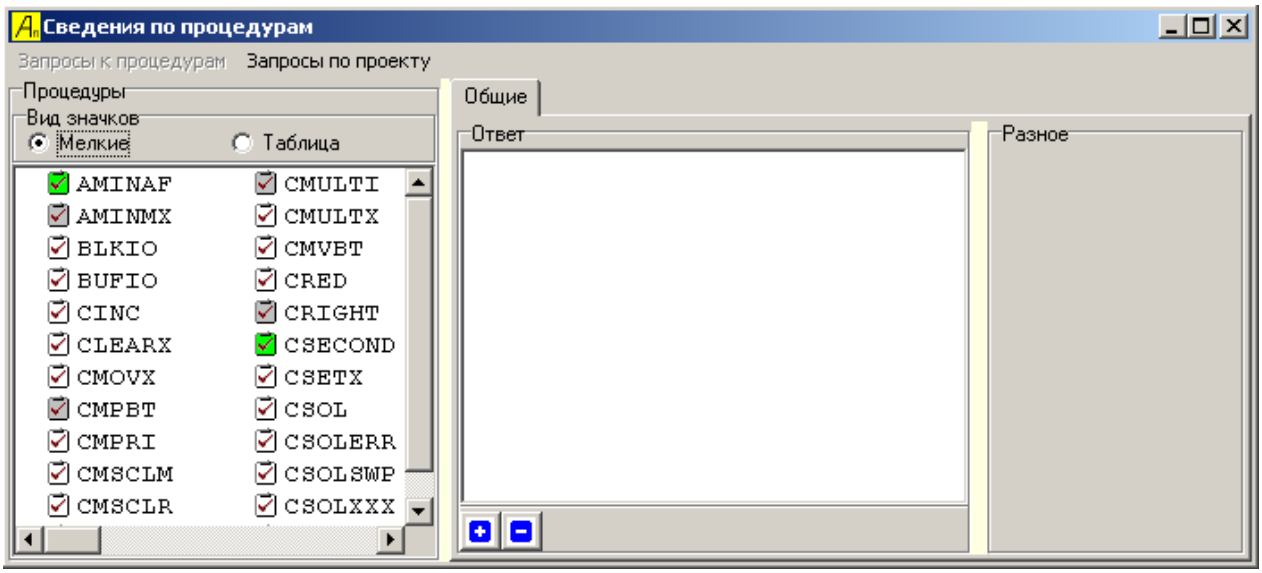

Рис. 7

Это окно имеет собственное меню, состоящее из 2-х функций:

- Запросы по проекту
- Запросы к процедурам

Окно разделено на две части. Одна часть (слева) содержит список всех программных единиц проекта. Список упорядочен по алфавиту. Главная процедура (**PROGRAM**) отмечена иконкой голубого цвета. Процедуры (**SUBROUTINE**) отмечены иконкой белого цвета, функции (**FUNCTION**) - иконкой зеленого цвета. Процедуры и функции, к которым в проекте нет обращения, отмечены иконкой серого цвета. В дальнейшем, всюду, где будут появляться имена процедур и функций, они будут сопровождаться этими иконками.

Вторая часть окна (справа) оформлена в виде блокнота с закладками и предназначена для вывода результатов запросов. Изначально блокнот имеет только одну закладку с именем **"Общие"** (которая существует всегда). На эту закладку в область **"Ответ"** выводятся ответы на запросы, относящиеся ко всему проекту в целом (см. Рис.7).

Запросы по проекту в целом сосредоточены в меню **"Запросы по проекту"** (см. Рис. 8):

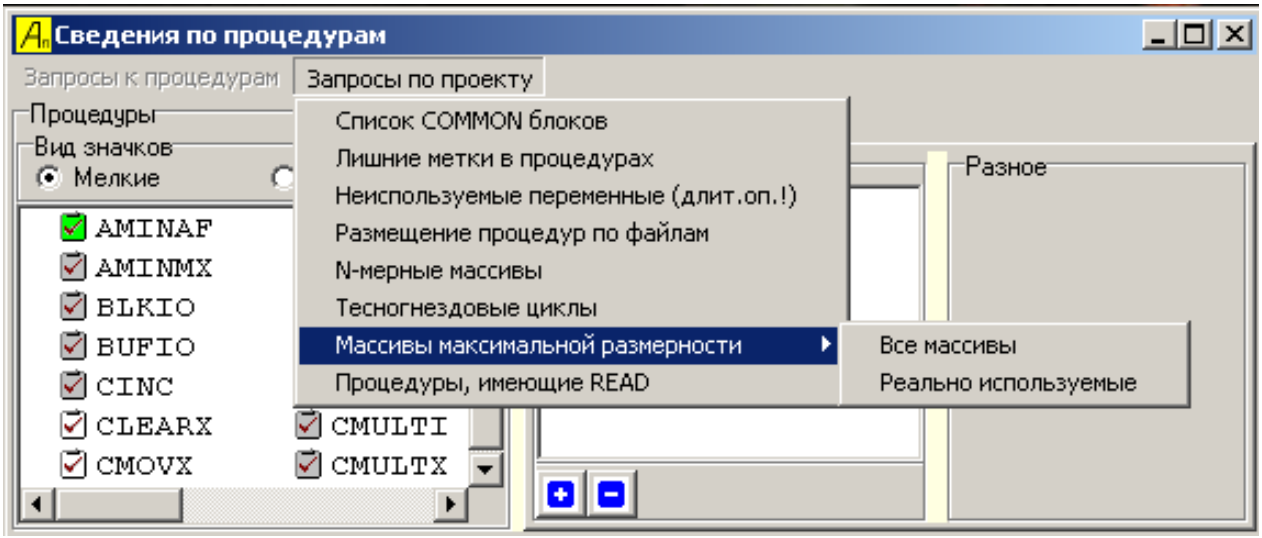

Рис. 8

Область **"Ответ"** содержит внизу 2 кнопки, которые управляют видом отображения: кнопка с обозначением  позволяет развернуть отображаемую схему, а кнопка с обозначением  позволяет свернуть отображаемую схему.

При выполнении запроса название области **"Ответ"** изменяется на соответствующее название запроса.

Другие закладки в блокноте появляются в ходе работы и предназначены для отображения сведений о конкретной программной единице. Для того чтобы появилась такая закладка, нужно щелкнуть (один раз) левой клавишей мыши на имени программной единицы в левой части окна. Выбранный таким образом объект отмечается дополнительной иконкой с красной точкой, а в

блокноте появляется закладка с именем выбранной процедуры. Можно выбрать несколько программных единиц, которым будут соответствовать несколько закладок в блокноте. Для того чтобы убрать ненужную закладку (или все закладки, кроме закладки **"Общие"**), надо в левой части окна указать курсором соответствующую программную единицу и щелчком правой клавишей мыши вызвать контекстное меню.

На Рис. 9 приведен пример с 3-мя выбранными процедурами (**DOPC**, **ERRO**, **GEOM**) и актуализировнной закладкой для процедуры **ERRO**.

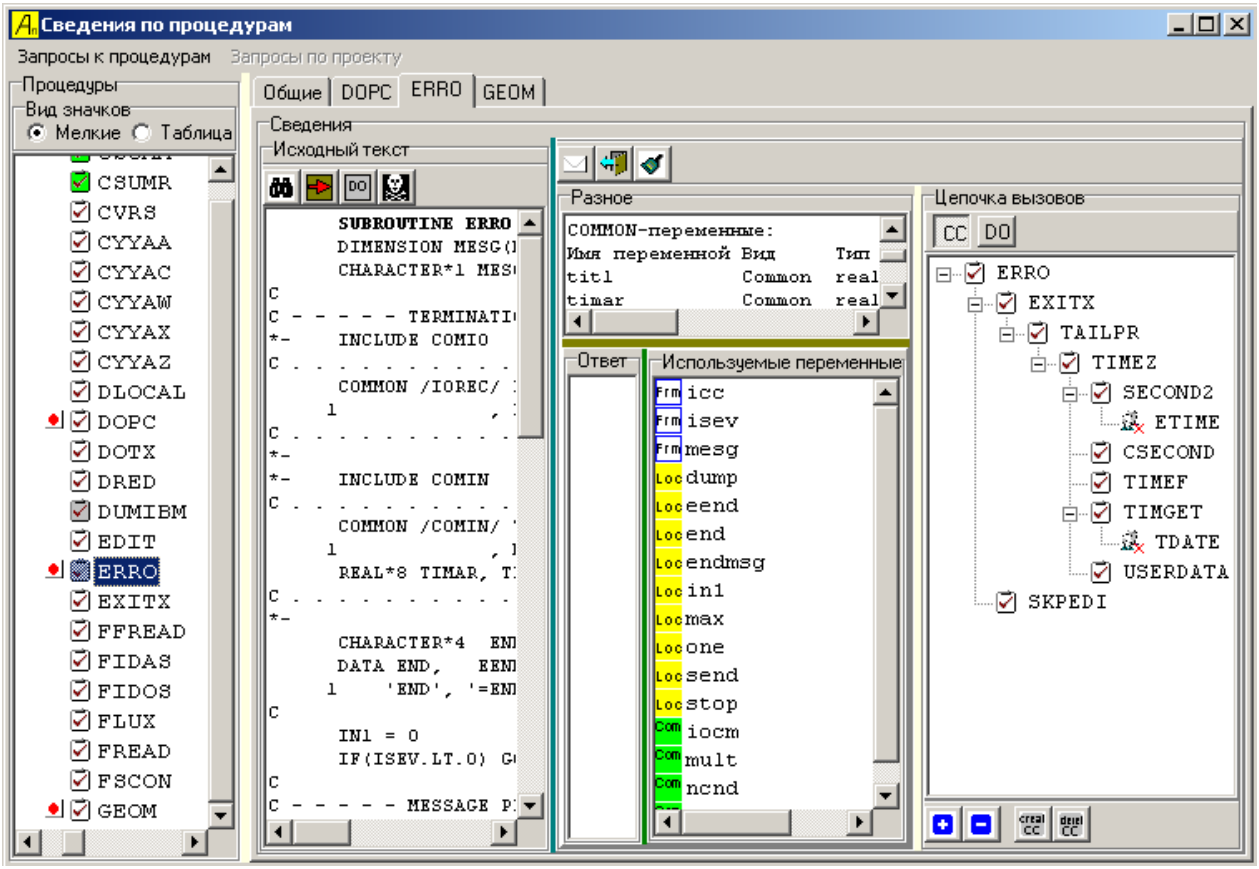

Рис. 9

На закладке для программной единицы присутствуют пять областей:

1. Область **"Исходный текст"**

Область расположена у левого края закладки. В ней отображается исходный текст программной единицы. Над этой областью находятся четыре функциональные кнопки.

- По кнопке - "Поиск текста" производится поиск заданной подстроки.
- Кнопка - "Расширить/Сузить область исходного текста" позволяет расширить область вправо за счет других областей или вернуть ее к исходному состоянию.
- По кнопке - "Отметить циклы DO" в тексте выделяются синим цветом строки, относящиеся к циклам DO, либо ко всем циклам, либо только к тесногнездовым. Вариант выбора задает пользователь.

- По кнопке  - "Найти мертвый код" красным цветом отмечаются строки недоступного кода.

2. Область **"Разное"**

Область расположена в средней части закладки сверху. При обработке некоторых запросов в нее выдаются ответы в текстовом виде одновременно с выдачей ответа в другую область в графическом виде (например, по поводу переменных). Над этой областью расположены три кнопки, управляющие видимостью областей:

- Кнопка  - "Скрыть/Показать дерево" управляет видимостью области **"Цепочка вызовов"**. Когда область **"Цепочка вызовов"** закрывается, ширина областей **"Разное"**, **"Ответ"** и **"Переменные"** увеличивается за счет освобожденного места.
- Кнопка  - "Скрыть/Показать область Разное" управляет видимостью области **"Разное"**. Когда область **"Разное"** закрывается, высота областей **"Ответ"** и **"Переменные"** увеличивается за счет освобожденного места.
- Кнопка  - "Очистить область Разное" удаляет содержимое области **"Разное"**.

3. Область **"Использованные переменные"**

Область расположена в центре закладки внизу справа, под областью **"Разное"**. В ней отображается в графическом виде список переменных, использованных в данной программной единице. Эти переменные отображаются всегда (без специального запроса). Переменные упорядочены по категориям и в пределах одной категории по алфавиту. Последовательность отображаемых категорий:

1. формальные параметры программной единицы,
2. именованные константы (PARAMETER),
3. локальные переменные,
4. глобальные (COMMON-переменные).

С каждой категорией переменной связана иконка определенного цвета:

- с формальным параметром - иконка белого цвета с текстом **Frm** 
- с именованной константой - иконка бирюзового цвета с текстом **Par** 
- с локальной переменной - иконка желтого цвета с текстом **Loc** 
- с COMMON переменной - иконка зеленого цвета с текстом **Com** 

В дальнейшем, везде, где будут появляться имена переменных, это соответствие иконок и категорий переменных будет соблюдаться. Переменные, имеющие синонимы, дополнительно отмечаются красной точкой внутри иконки  .

4. Область **"Ответ"**

Область расположена в центре закладки внизу слева, под областью **"Разное"**. В эту область помещается ответ на запрос, выбранный с помощью функций меню **Запросы к процедурам**. Заголовок области - **"Ответ"** существует только до первого запроса. Когда в окно помещается ответ на запрос, заголовок изменяется так, чтобы он соответствовал запросу.

5. Область **"Цепочка вызовов"** и **"Схема циклов DO"**

Область расположена у правого края закладки. Переключение между цепочкой вызовов и схемой циклов производится кнопками, расположенными над этой областью: при нажатии кнопки CC отображается цепочка вызовов, при нажатии кнопки DO - иерархическая схема циклов. Пример цепочки вызовов для процедуры **ERRO** приведен на рис.9., а пример схемы циклов DO приведен на Рис.10. Под этой областью располагаются еще четыре кнопки, из ко-

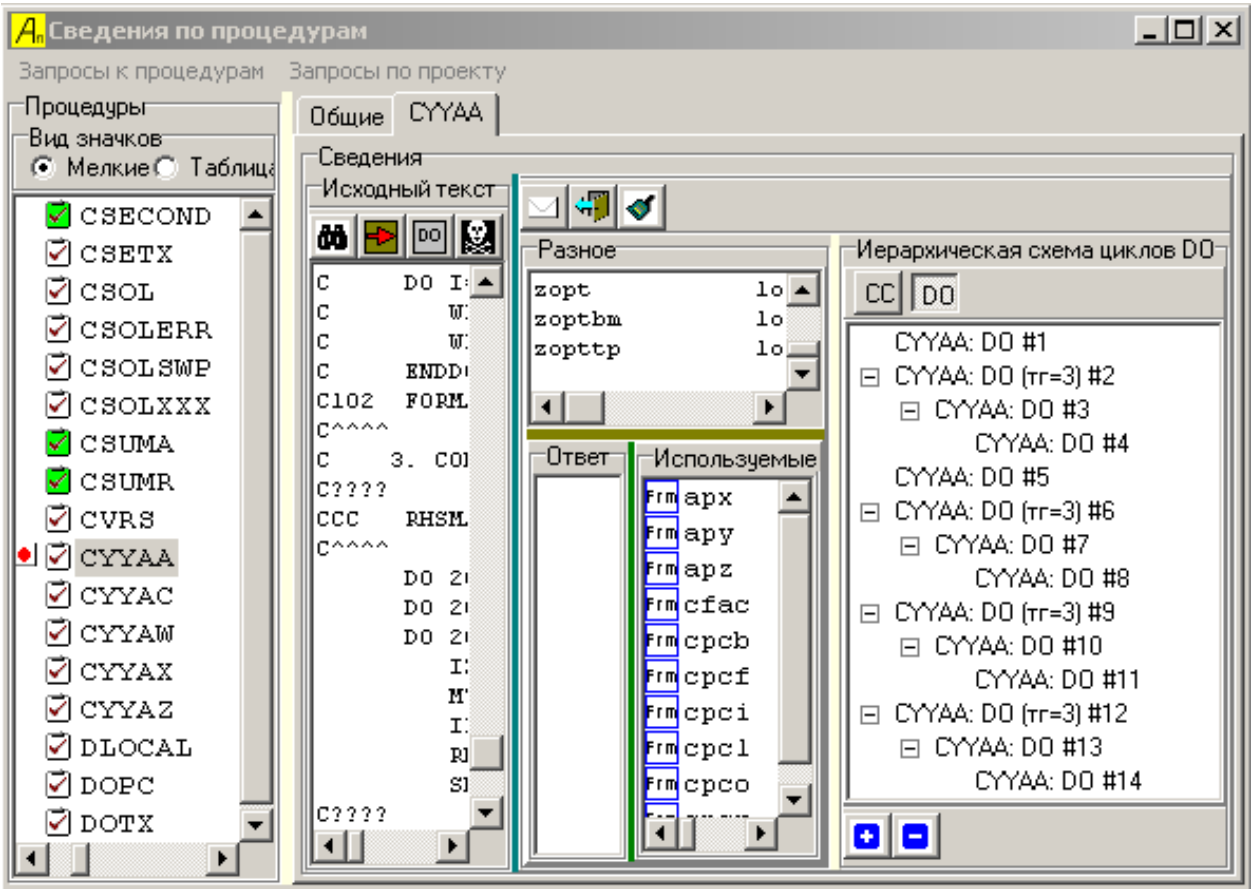

Рис. 10

торых две управляют видом отображения (кнопка с обозначением ⊞ означает развернуть отображаемую схему, а кнопка с обозначением ⊟ означает свернуть отображаемую схему) и две кнопки служат для удаления и повторного создания цепочки вызовов процедур. (Две последние кнопки видны только при нажатой кнопке CC.) Перед повторным созданием цепочки вызовов можно с помощью меню **Настройка** в главном окне программы задать условия, которые будут учтены при построении цепочки вызовов. Отметим,

что на Рис. 9 программные единицы ETIME и TDATE отмечены иконками особого вида . Это показывает обращение к программным единицам, тела которых в проекте не описаны.

Циклы DO с номерами 2, 6, 9 и 12, несут дополнительную информацию – (тг=3). Эта пометка означает, что соответствующее предложение DO относится к внешнему циклу тесно вложенного гнезда, имеющего глубину равную трем (см. Рис. 10).

Размеры областей на закладке могут быть изменены (за счет соседних), используя буксировку границ области мышью.

3.2 Пункты меню “Запросы по проекту”

3.2.1 Список COMMON-блоков

При обращении к этой функции в области ответов отображается список имен всех COMMON-блоков, упомянутых в проекте (см. Рис. 11).

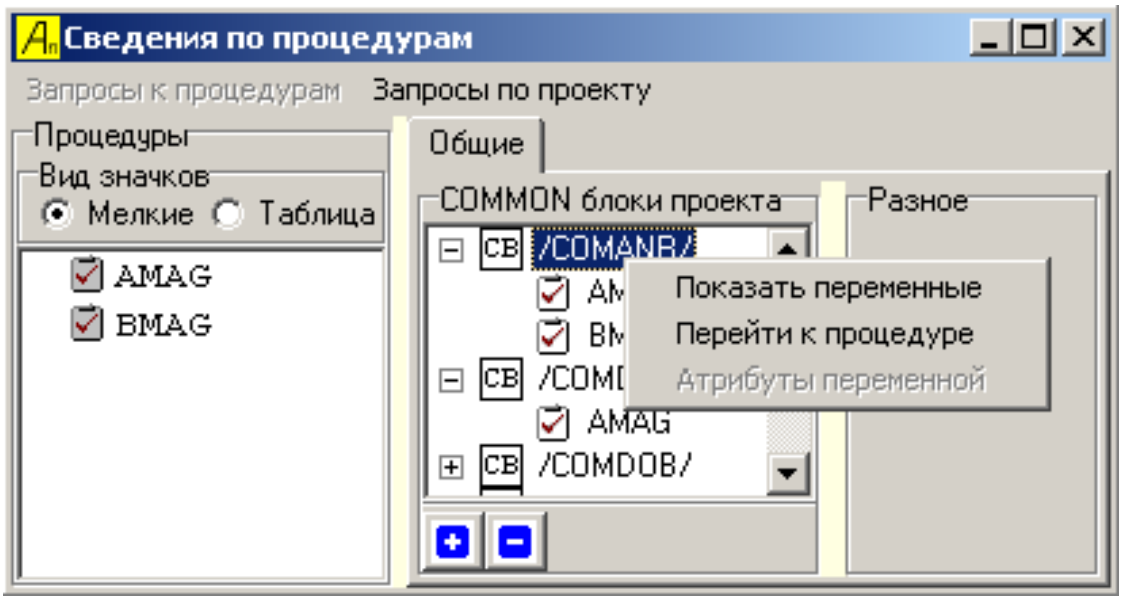

Рис. 11

Чтобы увидеть список программных единиц, в которых этот COMMON-блок объявлен, достаточно раскрыть соответствующую строку (щелчок на иконке ⊞). С элементами этого списка связано контекстное всплывающее меню, которое позволяет получить информацию о переменных COMMON-блока выбранной программной единицы, или перейти к исследованию этой программной единицы.

При выборе функции **Показать переменные** соответствующая информация выдается в специальном всплывающем окне (см. Рис. 12). Из первой строки окна, приведенного на рисунке видно, что переменная **vr** представляет собою одномерный массив с границами 1:**ndimuz** элементы которого имеют тип **DOUBLE PRECISION**. Тип задан предложением **IMPLICIT**. (Переменная **ndimuz**, задающая верхнюю границу массива, это - именованная константа, заданная предложением **PARAMETER**.)

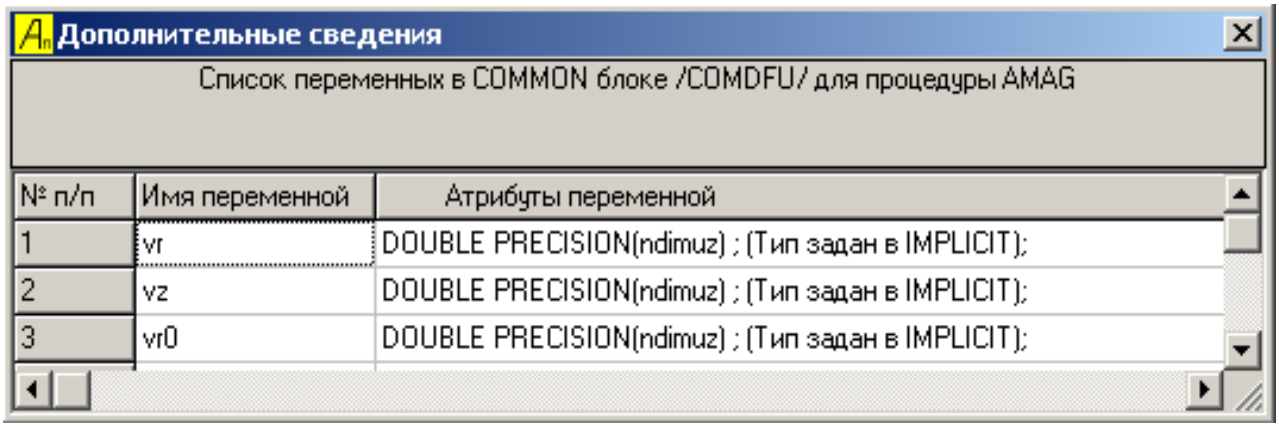

Рис. 12.

Если в описанной на Рис. 11 ситуации в контекстном всплывающем меню выбрать функцию **Перейти к процедуре**, то будет открыта закладка, соответствующая указанной программной единице. Если такой закладки в блокноте не было, то она будет создана.

3.2.2 Лишние метки в процедурах

При вызове этой функции в области ответов выдаются список программных единиц, содержащих метки, на которые нет ссылок.

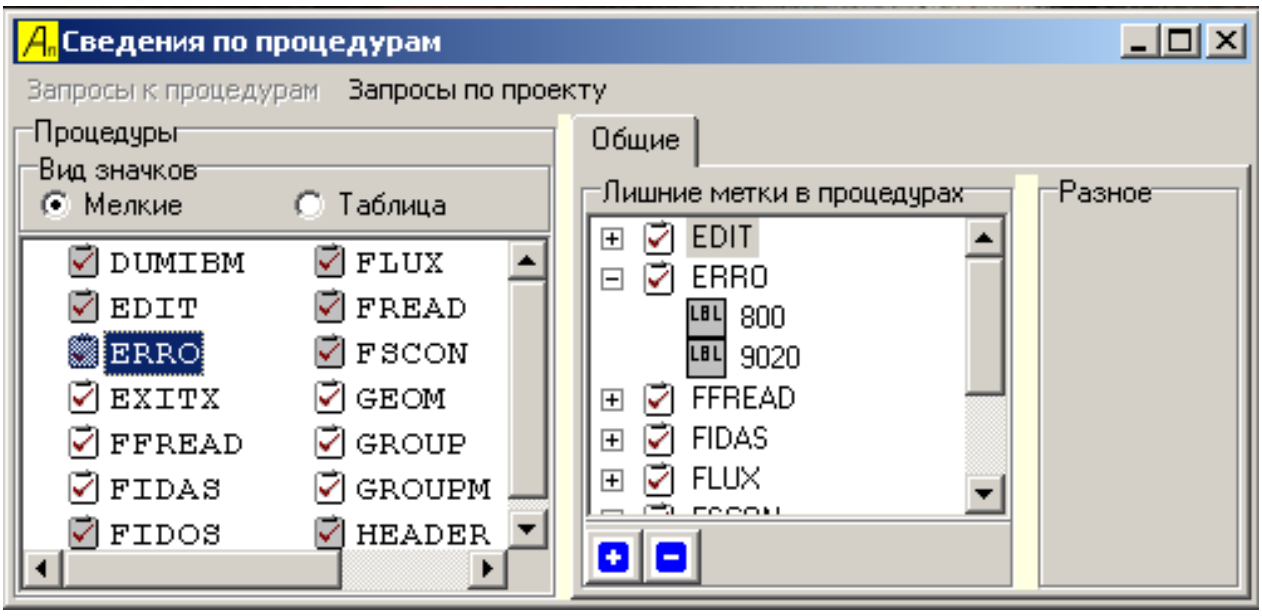

Рис. 13

При раскрытии строки с именем программной единицы (щелчке на иконке ⊞) под строкой отображается список лишних меток в указанной программной единице (см. Рис. 13). С именем программной единицы связано контекстное меню, позволяющее быстро перейти к соответствующей процедуре (функции), т.е. породить закладку, содержащую информацию о соответствующей программной единице, если ее раньше не было, или раскрыть ранее созданную закладку.

3.2.3 Неиспользуемые переменные

При вызове этой функции в области ответов отображается список программных единиц, в которых обнаружены неиспользуемые переменные. При раскрытии строки списка с именем программной единицы (щелчке на иконке ⊞) под строкой раскрывается список неиспользуемых в данной программной единице переменных (см. Рис. 14).

Контекстное всплывающее меню позволяет быстро перейти к выбранной программной единице.

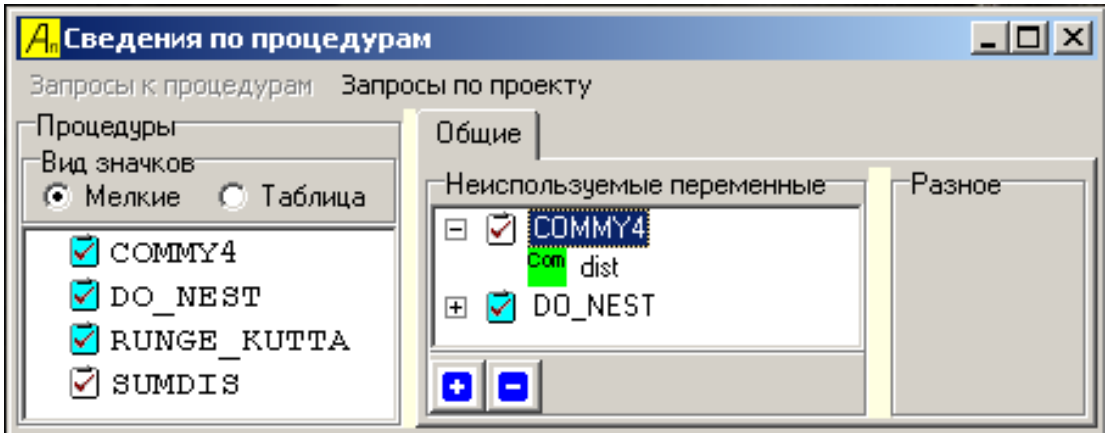

Рис.14

3.2.4 Размещение процедур по файлам

В области **"Процедуры"** список программных единиц упорядочен по имени. Для того чтобы понять, в каком файле находится соответствующая программная единица, надо обратиться к данной функции. При использовании этой функции в области ответов отображается список файлов, входящих в проект (см. Рис. 15). При раскрытии строки с именем файла (щелчке на иконке ⊞) под строкой отображается список программных единиц, описанных в соответствующем файле. Для каждого элемента списка программных единиц доступно контекстное меню, позволяющее быстро перейти к выбранной программной единице.

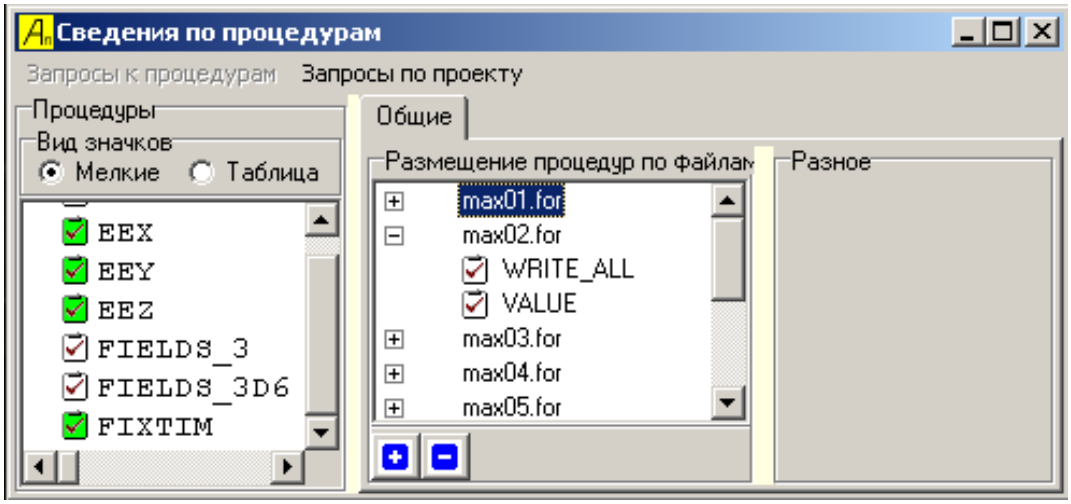

Рис. 15

3.2.5 N-мерные массивы

При обращении к этой функции открывается дополнительное модальное окно, в котором пользователь должен указать минимальную и максимальную размерность (количество граничных пар) интересующих его массивов. После задания интервала размерностей в области ответов перечисляется множество массивов, удовлетворяющих критерию запроса (см. Рис. 16). При раскрытии строки, соответствующей некоторому массиву (щелчок на иконке ⊞), под строкой отображается список программных единиц, в которых такой массив описан. С элементами списка программных единиц связано контекстное меню, позволяющее быстро перейти к указанной программной единице.

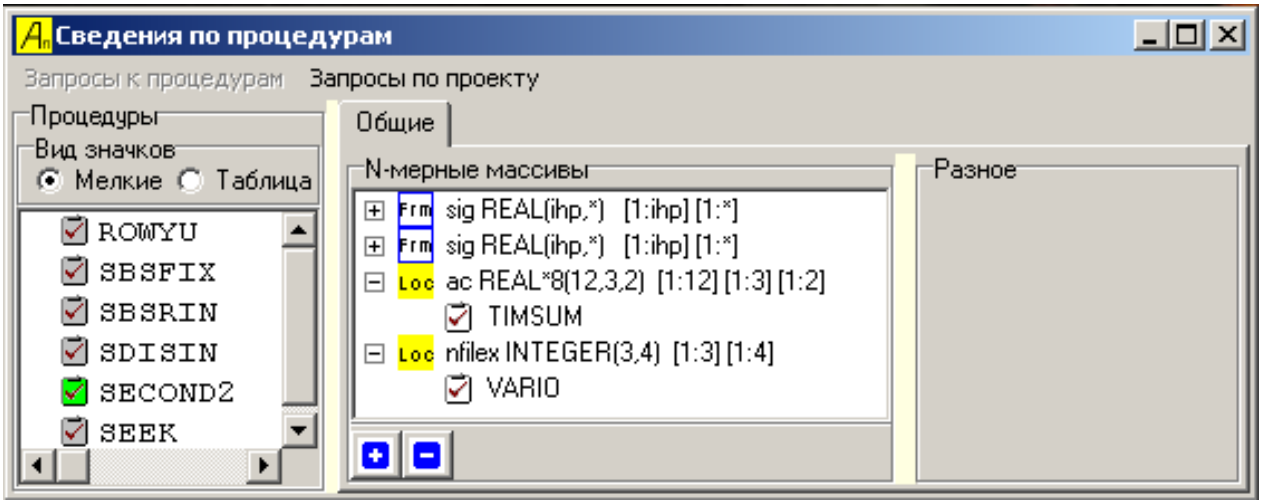

Рис. 16

3.2.6 Тесногнездовые циклы

При вызове этой функции просматриваются все программные единицы, входящие в проект, с целью обнаружения в них тесногнездовых циклов. Результаты поиска отображаются в области ответов в виде списка программных единиц, содержащих такие циклы. При раскрытии строки (щелчок на иконке ⊞) под ней выдается дополнительная информация о количестве тесногнездовых циклов, обнаруженных в данной программной единице.

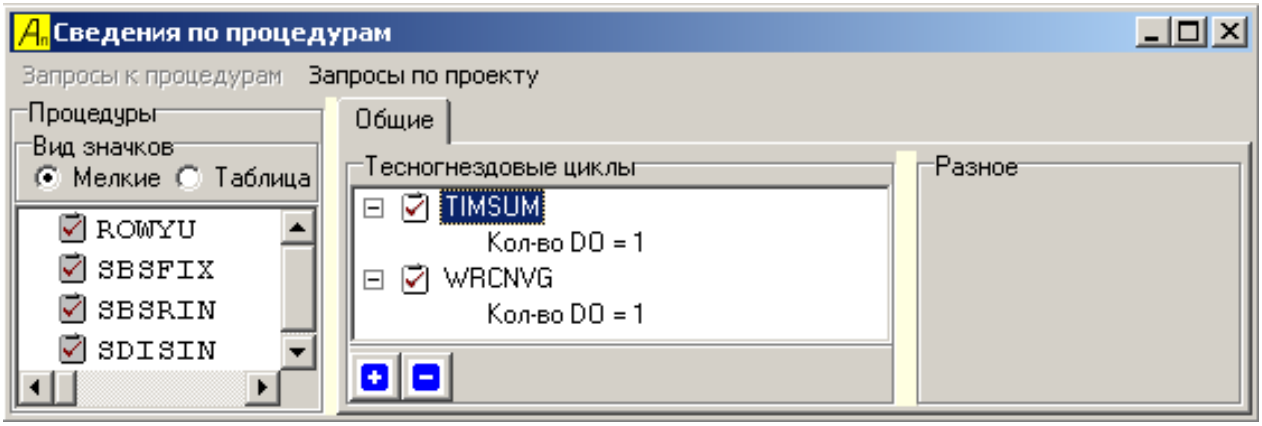

Рис. 17

В примере, приведенном на Рис. 17, в программных единицах TIMSUM и WRCNVG обнаружено по одному тесногнездовому циклу. Из этого окна также возможен быстрый переход к закладке, соответствующей программной единице, где доступна дополнительная информация о циклах DO.

3.2.7 Массивы максимальной размерности

При вызове этой функции во всех программных единицах просматриваются объявления массивов. Обнаруженные объявления заносятся в список. Затем в списке отыскиваются массивы максимальной размерности, которые и отображаются в области ответов в форме, аналогичной приведенной на Рис. 16. Возможны два режима отображения:

- **Все массивы** – отображаются массивы найденные в объявлениях программных единиц .
- **Реально используемые** – отображаются только те массивы, которые используются в исполняемых предложениях программных единиц.

3.2.8 Процедуры, имеющие READ

При вызове функции просматриваются все программные единицы, входящие в проект, с целью обнаружения в них предложений READ. Результаты поиска отображаются в области ответов в виде списка программных единиц (см. Рис. 18).

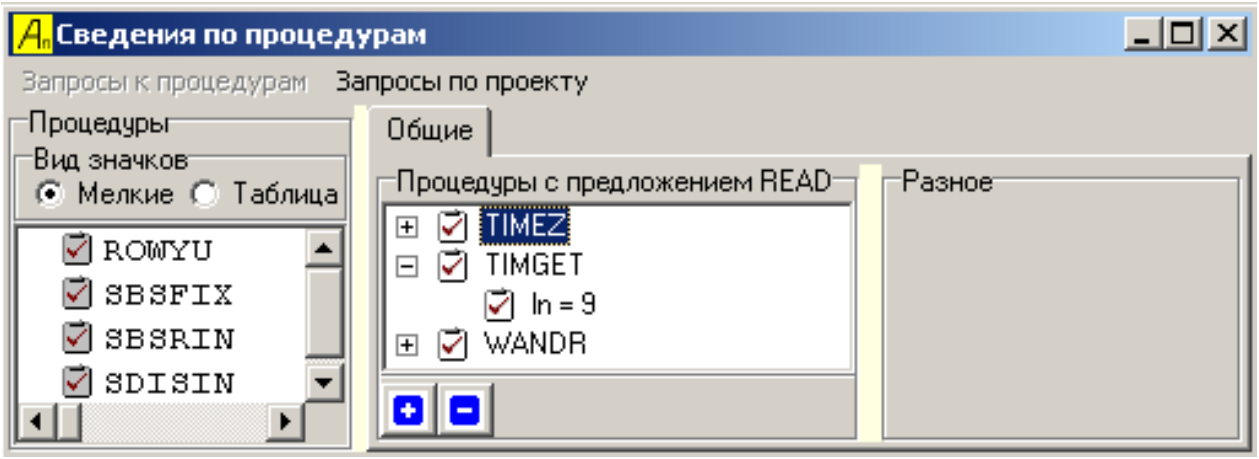

Рис.18

С элементами этого списка связано контекстное меню, позволяющее быстро перейти к исследованию указанной программной единицы.

3.3 Пункты меню “Запросы к процедурам”

3.3.1 Вызываемые процедуры

Функция просматривает предложения программной единицы с целью обнаружения обращений из данной программной единицы к другим программным единицам. Результаты поиска отображаются в области ответов.

3.3.2 Список COMMON блоков

Функция выполняется так же, как аналогичная функция в разделе запросов по проекту, с той лишь разницей, что во-первых, отображается список только тех COMMON-блоков, которые объявлены в данной программной единице, во-вторых, отображение производится в области ответов соответствующей закладки в виде перечня имен переменных с иконками.

3.3.3 Список обращений

Функция просматривает все программные единицы, входящие в проект, с целью обнаружения обращений к данной программной единице. Результаты поиска отображаются в области ответов и одновременно в области **"Разное"** (см. Рис.19).

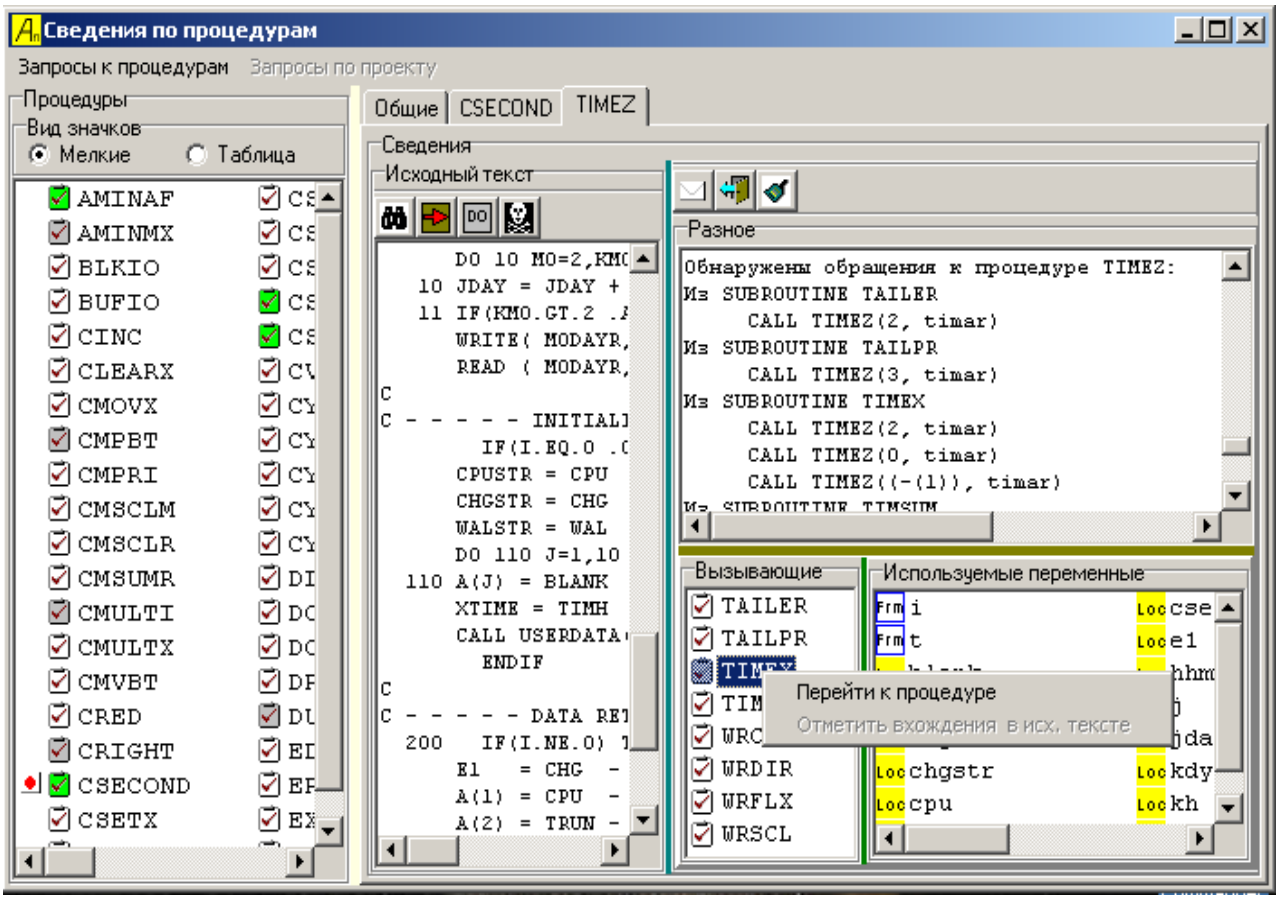

Рис. 19

Контекстное меню, связанное с элементом списка вызывающих программных единиц, позволяет быстро перейти к выбранной единице.

3.3.4 Список меток

Функция просматривает предложения программной единицы с целью обнаружения помеченных предложений. Список найденных меток она помещает в область ответа. (см. Рис.20) С каждой меткой в окне ответов связано контекстное меню, позволяющее отметить все упоминания данной метки в тексте программной единицы в области **"Исходный текст"**.

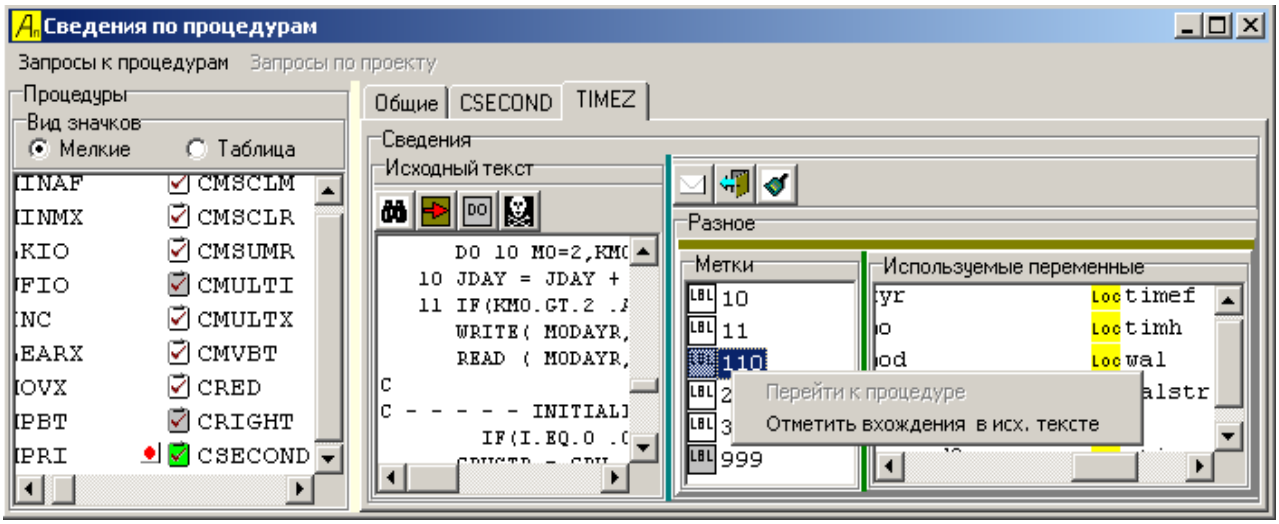

Рис. 20

3.3.5 Лишние метки

Функция просматривает предложения программной единицы с целью обнаружения помеченных предложений, на метки которых нет ссылок. В случае обнаружения так их предложений список меток помещается в область ответа, аналогично изложенному выше.

3.3.6 N-мерные массивы

Функция позволяет обнаружить в программной единице те массивы, размерность которых (количество граничных пар) находится в заданных пределах. При вызове этой функции пользователю предъявляется модальное окно, в котором ему предлагается задать минимальное и максимальное значение размерности. Список массивов, отвечающих заданному критерию, помещается в область ответов в виде перечня имен переменных с иконками разновидностей.

3.3.7 Неиспользуемые переменные

Функция обнаруживает переменные, объявленные в данной программной единице, но не использованные в исполняемых предложениях, и отображает их в области ответа на закладке данной программной единицы в виде перечня имен переменных с иконками разновидностей. (Используемые переменные отображаются в своей области всегда.)

3.4 Контекстные меню

С рядом областей окна **"Сведения по процедурам"** связано свое всплывающее меню, вызываемое щелчком правой кнопки мыши на соответствующем объекте. Частично они уже были описаны выше.

1) В области **"Процедуры"** в контекстном меню, связанном с графическим элементом, соответствующим программной единице доступны 2 функции:

- **Убрать выделение элемента**
- **Убрать выделение всех элементов**

Первая функция удаляет в блокноте закладку, соответствующую выделенной программной единице. Вторая функция удаляет закладки всех ранее выделенных программных единиц. Закладка **"Общие"** не удаляется.

2) В области ответа на закладке **"Общие"** во всплывающем меню имеется 3 функции:

- **Показать переменные**
- **Перейти к процедуре**
- **Атрибуты переменной**

Функция **Показать переменные** актуальна только для запроса **Список COMMON-блоков**. В окне ответов нужно раскрыть COMMON-блок, выделить процедуру и выполнить для нее эту функцию. В результате появится окно с заголовком **"Дополнительные сведения"**, в котором отобразится список переменных для соответствующего COMMON блока и соответствующей процедуре.

Функция **Перейти к процедуре** актуальна для всех запросов и позволяет вызвать закладку для выделенной в окне ответов процедуры и перейти на эту закладку.

Функция **Атрибуты переменной** актуальна для запроса **Неиспользуемые переменные** и показывает атрибуты выделенной в окне ответов переменной.

3) В области ответа на закладке для процедуры во всплывающем меню имеется 2 функции:

- **Перейти к процедуре**
- **Отметить вхождения в исх. тексте**

Функция **Перейти к процедуре** актуальна для запросов **Вызываемые процедуры** и **Список обращений** и позволяет вызвать закладку для выделенной в окне ответов процедуры и перейти на эту закладку.

Функция **Отметить вхождения в исх. тексте** актуальна для запросов: **Список меток** и **Лишние метки**, и отмечает в исходном тексте вхождения соответствующего метке текста.

4) В области **"Используемые переменные"** на закладке для процедуры во всплывающем меню имеется 5 функций (Рис. 21):

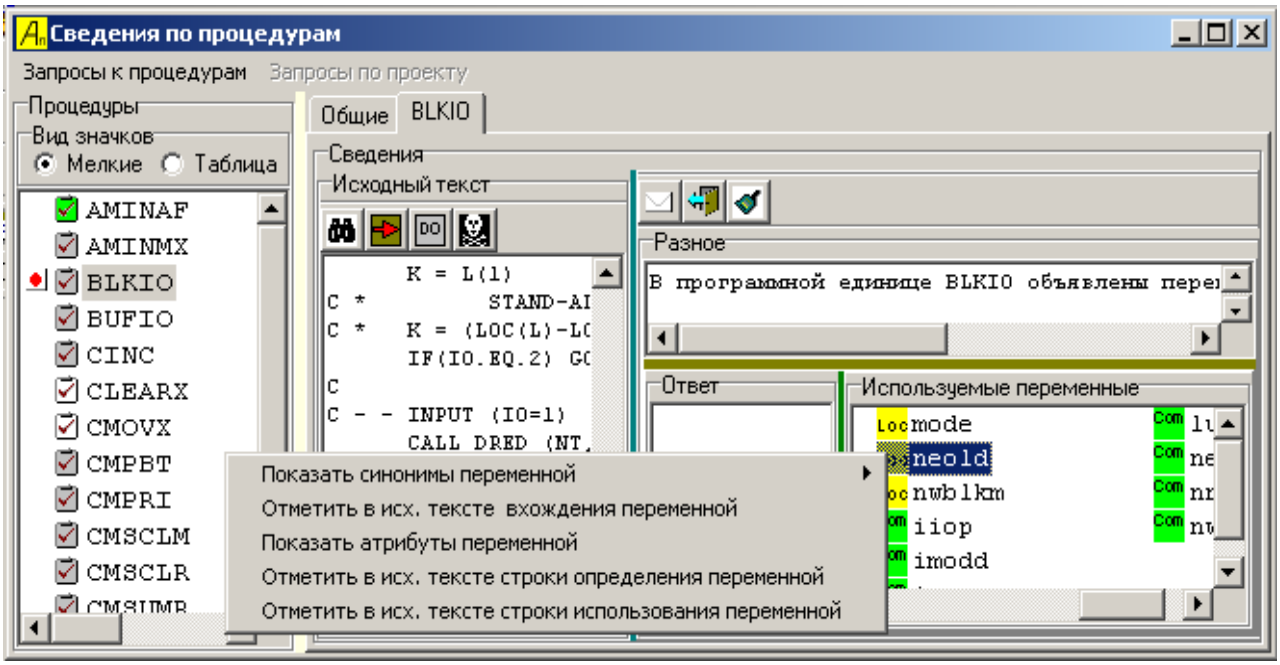

Рис. 21

- **Показать синонимы переменной**
  - **Все**
  - **Отличающиеся**
- **Отметить в исх. тексте вхождения переменной**
- **Показать атрибуты переменной**
- **Отметить в исх. тексте строки определения переменной**
- **Отметить в исх. тексте строки использования переменной**

Функция **Показать синонимы переменной** (**Все** или **Отличающиеся**) имеет смысл только для переменных, отмеченных иконкой с красной точкой. Список синонимов выдается в окно ответов и дублируется в окне **"Разное"**. Для остальных переменных (у которых нет синонимов) в окне **"Разное"** выдается сообщение: **"Синонимы не найдены"**.

Анализатор различает следующие виды синонимов:

- Глобальные синонимы. Если в разных программных единицах объявлены одноименные COMMON-блоки, и в этих COMMON-блоках в одинаковых позициях находятся разные имена, то такие имена рассматриваются как синонимы. (В одноименных COMMON-блоках переменные отождествляются по их позиции в блоке).
- Локальные синонимы. Это имена, перечисленные в списке EQUIVALENCE. В таком списке могут появляться, также, и имена COMMON-переменных.

- Временные синонимы. Поскольку в языке Фортран в качестве фактических параметров передаются адреса объектов, то в случае, когда фактическим параметром является ссылка на переменную, формальный параметр соответствующей программной единицы является квази синонимом такой переменной на время обслуживания обращения.

Функция **Отметить в исх. тексте вхождения переменной** отмечает в окне **"Исходный текст"** вхождения соответствующего имени переменной текста

Функция **Показать атрибуты переменной** отображает атрибуты переменной. Для всех видов переменных указывается тип переменной и способ его задания (явный, при помощи декларации IMPLICIT или по умолчанию). Для именованных констант (PARAMETER) дополнительно приводится вычисленное значение. Для формальных параметров – их позиция в списке. Для COMMON-переменных – имя COMMON-блока и позиция переменной в блоке.

Функция **Отметить в исх. тексте строки определения переменной** отмечает в исходном тексте зеленым цветом строки, в которых производится запись значения переменной.

Функция **Отметить в исх. тексте строки использования переменной** отмечает в окне **"Исходный текст"** фиолетовым цветом строки, в которых производится использование переменной.

5) В области отображения цепочки вызовов процедур во всплывающем меню имеется 2 функции:

- **Нарисовать граф**
- **Печатать цепочку вызовов**

Функция **Нарисовать граф** в отдельном всплывающем окне отображает цепочку вызовов в графическом виде с помощью графического визуализатора [4]. Пример графического изображения процедуры **ERRO** (см. Рис. 9) приведен в Приложении 2.

Функция **Печатать цепочку вызовов** позволяет выдать цепочку вызовов, на принтер.

6) В области отображения схемы циклов DO в контекстном меню доступны 3 функции:

- **Показать строку в исходном тексте**
- **Переменные в теле DO**
- **Печать иерархической схемы DO**

Функция **Показать строку в исходном тексте** выделяет в окне исходного текста программной единицы синим цветом строку, содержащую соответствующее предложение DO.

Функция **Переменные в теле DO** отображает во всплывающем окне с заголовком **"Дополнительные сведения"** список переменных, встретившихся в теле выбранного цикла DO, и дополнительные сведения об индексах этих переменных.

Ном.стр.	Переменная	Тип испо	Ном. инд	Тип инд.	Инд. пере	Текстовы	Нач.зн.	Кон. зн.	Шаг
5	K	W							
6	J	W							
7	I	W							
8	Y	W	0	2	I	I	5	50	1
8	Y	W	1	2	J	J	-10	20	1
8	Y	W	2	2	K	K	1	50	1
8	X	R	0	2	I	I	5	50	1
8	X	R	1	3		J + 5	-10	20	1
8	X	R	2	4		7 * K + 11	1	50	1

Рис. 22

Рис. 22 соответствует приведенной ниже программе:

```
PROGRAM DO_NEST
REAL X(100,101,102),Y(-50:50,-20:80,-25:75)
PARAMETER (KMAX=50,JMAX=20,IMAX=50)
INTEGER I,J,K
DO K=1,KMAX
   DO J=-10,JMAX
      DO I=5,IMAX
         Y(I,J,K)=X(I,J+5,7*K+11)
      ENDDO
   ENDDO
ENDDO
END
```

В первом столбце окна показан номер строки исходного текста, в которой находится рассматриваемая переменная.

Во втором столбце приведено имя переменной.

В третьем столбце показан способ использования переменной. (W – на запись, R – на чтение).

В четвертом столбце указан порядковый номер индекса. (Индексы переменной нумерованы, начиная с 0). Для доступа к переменным X и Y использовано три индекса.

В пятом столбце показан тип индексного выражения. В анализаторе использовано 5 типов индексных выражений:

1. – константа,
2. – простая переменная,
3. – переменная со смещением,
4. – линейное выражение,
5. – все остальное.

В шестом столбце показана соответствующая переменная цикла DO.

В седьмом столбце – индексное выражение в текстовом виде.

В остальных столбцах показаны начальное и конечное значение переменной цикла и шаг ее изменения, если эти значения доступны (заданы константными выражениями) и если в индексное выражение входит только одна переменная цикла.

Функция **Печатать иерархической схемы DO** позволяет распечатать структуру циклов DO, используя принтер.

Для областей **"Исходный текст"** и **"Разное"** контекстные меню не предусмотрены.

4. Состав анализатора и вспомогательные программы.

Для установки программы необходимы следующие файлы:

1) Основная программа и системные библиотеки, которые могут располагаться в любом каталоге по усмотрению пользователя:

Tool_O_pro.exe
borlndmm.dll
vcl.lib
cg32.dll
cc3250mt. dll

2) Дистрибутив графического визуализатора graphviz-1.11.exe [8], который устанавливает в каталоге Program Files подкаталог ATT с графическим визуализатором.

3) Вспомогательная утилита Includer.exe для вставки содержимого INCLUDE-файлов в тело программы. Эта утилита использует те же библиотеки, что и основная программа, поэтому она должна располагаться в том же каталоге.

Заключение

В данной работе описана первая попытка создания удобного инструмента анализа программ. Нам представляется, что использованный подход позволяет получать много полезных сведений о больших программах. Ограниченность ресурсов и стремление скорее получить практический результат обусловили использование для интерфейса с пользователем встроенных в C++Builder визуальных элементов графического диалога. Дальнейшая работа над интерфейсом могла бы его улучшить.

Глубину анализа программ необходимо увеличить, особенно в направлении сбора зависимостей по данным.

Некоторые операции анализатора, особенно когда они выполняются над проектом в целом, занимают значительное время. Возможно, надо уделить больше внимания эффективности их реализации.

Авторы считают своим долгом выразить благодарность д. ф.-м. н. Андрианову А.Н., к. ф.-м. н. Головкову С.Л. и к. ф.-м. н. Ефимкину К.Н. за предоставленные большие программы, на которых отлаживался анализатор, а также за проявленное к данной работе внимание и полезные обсуждения.

Литература

1. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления – СПб.: БХВ-Петербург, 2002.
2. S.Johnson, H.Jin and C.Ierotheou. “The Para WiseExpert Assistant – Widening Accessibility to efficient, Scalable Tool Generated Open MP Code”. Proceedings of WOMPAT 2004, (2004). (www.parallelsp.com)
3. C. Ierotheou, S.P. Johnson, M. Cross, P.F. Leggett, “Computer Aided Parallelisation Tools (CAPTools) – conceptual overview and performance on the parallelisation of structured mesh codes”, *Parallel Computing,* Vol. 22, pp 163-195, 1996.
4. http://polaris.cs.uiuc.edu/newhome.
5. www-suif.stanford.edu/research.
6. Т.П. Баранова, В. Ю. Вершубский. Использование библиотеки классов пакета “SAGE” для анализа программ, написанных на языке Фортран. Препринт Института прикладной математики им. М. В. Келдыша РАН № 69, Москва, 2004, 30 стр.
7. www.extreme.indiana.edu
8. www.graphviz.org

Приложение 1. Пример INI-файла

Раздел состоит из заголовка - названия раздела, заключенного в квадратные скобки, - и набора ключевых параметров.

```
[LOG_FILE]
isLogFile=0
[SUBROUTINE_CHANNEL]
LibFunc=1
MPIFunc=1
[SHOW_SUBROUTINE]
SUBROUTINE1=YES
SUBROUTINE2=NO
     . . .
SUBROUTINE=YES
```

Раздел [LOG_FILE] и его параметр в настоящее время не используются и являются заделом для последующего развития программы анализатора.

Раздел [SUBROUTINE_CHANNEL] содержит параметры:

LibFunc	1	библиотечные функции будут исключаться
	0	библиотечные функции будут включаться
MPIFunc	1	MPI- функции будут исключаться
	0	MPI- функции будут включаться

Раздел [SHOW_SUBROUTINE] содержит список:

<имя процедуры>=YES | NO

YES - процедура учитывается,

NO - процедура не будет учитываться.

Приложение 2. Пример графа вызовов для процедуры ERRO