Подсистема визуализации поиска решения на графе состояний в РДО

Содержание

Перечень сокращений

Терминология

1. Введение
2. Предпроектное исследование
   1. Основные положения языка РДО
   2. Поиск решения на графе пространства состояний
   3. Программный комплекс RAO-XT
   4. Постановка задачи
3. Формирование ТЗ
   1. Введение
   2. Общие сведения
   3. Назначение разработки
   4. Требование к программе или программному изделию
      1. Требования к функциональным характеристикам
      2. Требования к надежности
      3. Условия эксплуатации
      4. Требование к составу и параметрам технических средств
      5. Требование к информационной и программной совместимости
      6. Требование к маркировке и упаковке
      7. Требование к транспортированию и хранению
   5. Требования к программной документации
   6. Стадии и этапы разработки
   7. Порядок контроля и приемки
4. Концептуальный этап проектирования
   1. Графический интерфейс
   2. Отображение статистики
      1. Вывод статистики по результатам поиска
      2. Вывод статистики по вершине
   3. Вызов окна интерфейса подсистемы из интерфейса модуля трассировки
   4. Модуль визуализации в системе RAO-XT
5. Технический этап проектирования
   1. Проектирование библиотечной части подсистемы визуализации
      1. Формирование древовидной структуры
      2. Чтение записи начала поиска
      3. Чтение записи раскрытия вершины
      4. Чтение записи порождения новой вершины,
      5. Чтение записи решения
      6. Чтение записи завершения поиска
   2. Проектирование графической части подсистемы визуализации
      1. Выбор графической библиотеки
      2. –
6. Рабочий этап проектирования
   1. Вывод графа на экран
      1. Привязка к трассировке
      2. Вывод графов для нескольких точек принятия решений типа search
   2. Расположение графа в окне
   3. Вывод статистики по поиску решения
   4. –
7. Апробирование разработанной системы в модельных условиях
8. Заключение

Список используемых источников

Список использованного ПО

Приложение А. Исходный код модели, использованной для тестирования модуля

Перечень сокращений

ИМ – Имитационное Моделирование

СДС – Сложная Дискретная Система

IDE – Integrated Development Environment (Интегрированная Среда Разработки)

Терминология

Плагин — независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и/или использования её возможностей.

Трассировка — получение информационных сообщений о работе приложения во время выполнения.

Сериализация — процесс перевода какой-либо структуры данных в последовательность битов.

1. Введение

Имитационное моделирование (ИМ)[1] на ЭВМ находит широкое применение при исследовании и управлении сложными дискретными системами (СДС) и процессами, в них протекающими. К таким системам можно отнести экономические и производственные объекты, морские порты, аэропорты, комплексы перекачки нефти и газа, ирригационные системы, программное обеспечение сложных систем управления, вычислительные сети и многие другие. Широкое использование ИМ объясняется тем, что размерность решаемых задач и неформализуемость сложных систем не позволяют использовать строгие методы оптимизации. Эти классы задач определяются тем, что при их решении необходимо одновременно учитывать факторы неопределенности, динамическую взаимную обусловленность текущих решений и последующих событий, комплексную взаимозависимость между управляемыми переменными исследуемой системы, а часто и строго дискретную и четко определенную последовательность интервалов времени. Указанные особенности свойственны всем сложным системам.

Проведение имитационного эксперимента позволяет:

1. Сделать выводы о поведении СДС и ее особенностях:

* без ее построения, если это проектируемая система;
* без вмешательства в ее функционирование, если это действующая система, проведение экспериментов над которой или слишком дорого, или небезопасно;
* без ее разрушения, если цель эксперимента состоит в определении пределов воздействия на систему.

1. Синтезировать и исследовать стратегии управления.
2. Прогнозировать и планировать функционирование системы в будущем.
3. Обучать и тренировать управленческий персонал и т.д.

ИМ является эффективным, но и не лишенным недостатков, методом. Трудности использования ИМ, связаны с обеспечением адекватности описания системы, интерпретацией результатов, обеспечением стохастической сходимости процесса моделирования, решением проблемы размерности и т.п. К проблемам применения ИМ следует отнести также и большую трудоемкость данного метода.

Интеллектуальное ИМ, характеризующиеся возможностью использования методов искусственного интеллекта и прежде всего знаний, при принятии решений в процессе имитации, при управлении имитационным экспериментом, при реализации интерфейса пользователя, создании информационных банков ИМ, использовании нечетких данных, снимает часть проблем использования ИМ.

Разработка интеллектуальной среды имитационного моделирования РДО выполнена в Московском государственном техническом университете (МГТУ им. Н. Э. Баумана) на кафедре "Компьютерные системы автоматизации производства". Причинами ее проведения и создания РДО явились требования универсальности ИМ относительно классов моделируемых систем и процессов, легкости модификации моделей, моделирования сложных систем управления совместно с управляемым объектом (включая использование ИМ в управлении в реальном масштабе времени) и ряд других, сформировавшихся у разработчиков при выполнении работ, связанных с системным анализом и организационным управлением сложными системами различной природы.

2. Предпроектное исследование

2.1 Основные положения языка РДО

Основные положения системы РДО могут быть сформулированы следующим образом:

* Все элементы СДС представлены как ресурсы, описываемые некоторыми параметрами. Ресурсы могут быть разбиты на несколько типов; каждый ресурс определенного типа описывается одними и теми же параметрами.
* Состояние ресурса определяется вектором значений всех его параметров; состояние СДС - значением всех параметров всех ресурсов.
* Процесс, протекающий в СДС, описывается как последовательность целенаправленных действий и нерегулярных событий, изменяющих определенным образом состояние ресурсов; действия ограничены во времени двумя событиями: событиями начала и событиями конца.
* Нерегулярные события описывают изменения состояния СДС, непредсказуемые в рамках продукционной модели системы (влияние внешних по отоношению к СДС факторов либо факторов, внутренних по отношению к ресурсам СДС). Моменты наступления нерегулярных событий случайны.
* Действия описываются операциями, которые представляют собой модифицированные продукционные правила, учитывающие временные связи. Операция описывает предусловия, которым должно удовлетворять состояние участвующих в операции ресурсов, и правила изменения состояния ресурсов в начале и в конце соответствующего действия.
* Множество ресурсов R и множество операций O образуют модель СДС.

2.2. Поиск решения на графе пространства состояний

В языке имитационного моделирования РДО помимо возможности использования для описания законов управления формализмов продукционных правил введены так называемые точки принятия решений, позволяющие осуществлять оптимальное управление.

Механизм точек принятия решений в языке имитационного моделирования РДО позволяет гибко сочетать имитацию с оптимизацией. Для этого используется поиск на графе состояний.

Примерами задач, которые решаются с использованием точек принятия решений, могут служить:

* Различные транспортные задачи (например, выбор последовательности объезда пунктов транспортным средством при минимуме пройденного пути, времени или стоимости).
* Задачи укладки грузов при минимизации занимаемого ими объема (в более общем случае – задачи размещения).
* Нахождение решения логических задач за минимальное число ходов (например, расстановка фишек в игре «Пятнашки»).
* Задачи теории расписаний (например, задачи определения последовательности обработки различных деталей на станках при минимизации времени обработки, либо обслуживания клиентов с минимумом отклонений от запланированного времени изготовления заказов и т. д.).

Граф — основной объект изучения математической теории графов, совокупность непустого множества вершин и наборов пар вершин (связей между вершинами).

Граф задается двумя множествами:

* – множество вершин графа. Каждой вершине ставится в соответствие состояние системы (база данных);
* – множество дуг. Каждой дуге , принадлежащей и соединяющей пару вершин, ставится в соответствие правило преобразования (продукционное правило). Если дуга направлена от вершины к вершине , то в данном случае будет являться вершиной-родителем, а – вершиной-потомком (преемником).

Маршрутом в графе называют конечную последовательность вершин, в которой каждая вершина (кроме последней) соединена со следующей в последовательности вершиной ребром.

Цепью называется маршрут без повторяющихся рёбер.

Циклом называют цепь, в которой первая и последняя вершины совпадают.

Граф называется связным, если для любых вершин существует путь из в .

Граф называется деревом, если он связный и не содержит нетривиальных циклов.

В графах, представляющих интерес для поиска, у каждой вершины должно быть конечное число вершин-преемников. С дугой может быть связана некоторая величина – стоимость дуги, она отражает затраты (в смысле заданного критерия оптимизации) применения соответствующего правила.

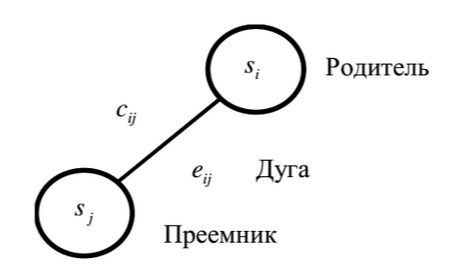
**

Рис 1. Простейший граф состояний

В графе имеются две особые вершины:

* – начальная вершина, или другими словами, вершина, представляющая собой исходную базу данных;
* – целевая вершина, иначе – вершина, представляющая собой базу данных, удовлетворяющую терминальному условию поиска. Таких вершин может быть не одна, а множество, и такое множество будет называться целевым множеством.

Путь в графе – последовательность вершин, в которой каждая последующая является преемником:

,

где – длина пути.

Каждому пути ставится в соответствие его стоимость, которая равна сумме стоимостей применения правил по всему пути на графе .

2.3. Программный комплекс RAO-XT

Программный комплекс RAO-XT предназначен для разработки и отладки имитационных моделей на языке РДО. Основные цели данного комплекса - обеспечение пользователя легким в обращении, но достаточно мощным средством разработки текстов моделей на языке РДО, обладающим большинством функций по работе с текстами программ, характерных для сред программирования, а также средствами проведения и обработки результатов имитационных экспериментов.

Система имитационного моделирования RAO-XT представляет собой плагин для интегрированной среды разработки Eclipse – свободной интегрированной среды разработки модульных кроссплатформенных приложений. Система написана на языке Java и состоит из трех основных компонентов:

* rdo – компонент, производящий преобразование кода на языке РДО в код на языке Java.
* rdo.lib – библиотека системы. Этот компонент реализует ядро системы имитационного моделирования.
* rdo.ui – компонент, реализующий графический интерфейс системы с помощью библиотеки SWT.

На момент начала выполнения курсового проекта, система не имела возможности выводить на экран пользователя граф пространства состояний для моделей, содержащих точки принятия решений. Вывод информации о поиске осуществлялся в текстовом формате в графическом окне модуля трассировки.

2.4 Постановка задачи

Проектирование любой системы начинается с выявления проблемы, для которой она создается. Под проблемой понимается несовпадение характеристик состояния систем, существующей и желаемой.

В результате предпроектного исследования было выявлено отсутствие в программном комплексе RAO-XT подсистемы графической визуализации поиска решения на графе пространства состояний.

Для повышения эффективности процесса моделирования была установлена необходимость разработать и внедрить в комплекс RAO‑XT такую подсистему.

3. Формирование ТЗ

3.1. Введение

~~Целью выполнения работ на стадии формирования ТЗ на проектируемую систему является уточнение и детализация требований заказчика, а также разработка требований к составу и содержанию работ по созданию системы, порядку приемки, документации, к составу и содержанию работ по подготовке проектируемой системы к эксплуатации.~~

3.2. Общие сведения

Основание для разработки: задание на курсовой проект.

Заказчик: Кафедра «Компьютерные системы автоматизации производства» МГТУ им. Н. Э. Баумана

Разработчик: студент кафедры «Компьютерные системы автоматизации производства» Стрижов К. А.

Наименование темы разработки: «Проектирование модуля визуализации поиска решения на графе состояний в РДО»

3.3. Назначение разработки

Разработать подсистему визуализации поиска решений на графе состояний и добиться ее интеграции в систему моделирования RAO-XT.

3.4. Требование к программе или программному изделию

3.4.1. Требования к функциональным характеристикам

Модуль визуализации должен удовлетворять следующим требованиям:

* Корректно отображать граф пространства состояний для соответствующей точки принятия решений типа search;
* Вызов графического окна модуля должен осуществляться из пользовательского интерфейса модуля трассировки системы RAO-XT;
* Графическое окно должно содержать всю необходимую информацию о точке принятия решений, как-то:
  + - Название точки принятия решений;
    - Статистика по поиску на графе состояний;
    - Подробная информация по выделенной вершине;
    - Выделение цветом вершин графа, относящихся к решению;
* Иметь возможность отображать несколько графов в случае наличия в модели нескольких точек принятия решений.

3.4.2. Требования к надежности

Основное требование к надежности направлено на поддержание в исправном и работоспособном состоянии ЭВМ, на которой происходит использование программного комплекса RAO-XT.

Сохранение работоспособности системы при отказе по любым причинам подсистемы или ее части.

3.4.3. Условия эксплуатации

Эксплуатация должна производиться на оборудовании, отвечающем требованиями к составу и параметрам технических средств, и с применением программных средств, отвечающим требованиям к программной совместимости.

Аппаратные средства должны эксплуатироваться в помещениях с выделенной розеточной электросетью 220В ±10%, 50 Гц с защитным заземлением.

3.4.4. Требование к составу и параметрам технических средств

Программный продукт должен работать на компьютерах со следующими характеристиками:

* объем ОЗУ не менее 2 Гб;
* объем жесткого диска не менее 50 Гб;
* микропроцессор с тактовой частотой не менее 1ГГц;
* монитор с разрешением от 1280\*1024 и выше.

3.4.5. Требование к информационной и программной совместимости

Система должна работать под управлением следующих ОС:

* Windows 7, 8;
* Ubuntu 14.10.

3.4.6. Требование к маркировке и упаковке

Требования к маркировке и упаковке не предъявляются.

3.4.7. Требование к транспортированию и хранению

Требования к транспортированию и хранению не предъявляются.

3.5. Требования к программной документации

Требования к программной документации не предъявляются.

3.6. Стадии и этапы разработки

Разработка должна быть проведена в три стадии:

* техническое задание
* технический и рабочий проекты
* внедрение

На стадии «Техническое задание» должен быть выполнен этап разработки и согласования настоящего технического задания.

На стадии «Технический и рабочий проект» должны быть выполнены перечисленные ниже этапы работ:

* разработка программы
* разработка методики тестирования (или удалить пункт или дописать записку)
* испытания программы

На стадии «Внедрение» должен быть выполнен этап разработки «Подготовка и передача программы».

3.7. Порядок контроля и приемки

Контроль и приемка работоспособности системы осуществляются с помощью следующих методов:

* Опрос экспертов. Используется для оценки дизайна, качества и удобства использования новой системы графического вывода;
* Многократное ручное тестирование с помощью имитационных моделей, написанных на языке РДО.

4. Концептуальный этап проектирования системы

На концептуальном этапе проектирования требовалось:

* разработать графический интерфейс окна вывода графа на экран;
* определить перечень необходимой информации, которую должна содержать выводимая на экран статистика;
* определить, как должен быть организован вызов окна графа из интерфейса модуля трассировки;
* разработать общую структуру подсистемы и схему его взаимодействия с другими компонентами системы RAO-XT.

4.1. Графический интерфейс

Разрабатываемая подсистема должна предоставлять пользователю графический интерфейс, содержащий в себе построенный граф пространства состояний и собранную по результатам поиска статистику. На данном этапе проектирования были приняты следующие решения:

* большую часть пространства вызываемого окна должен занимать построенный граф пространства состояний, поскольку он является главным предметом реализации интерфейса;
* построенный граф должен содержать минимум необходимой информации, как-то:
  + номера построенных вершин – они отображают последовательность построения графа пространства состояний;
  + информацию о решении (согласно требованиям, указанным в п. 3.4.1);
* вывод статистики должен быть компактным и не отнимать пространство у отображенного графа.

Предполагается, что изначально интерфейс предоставляет пользователю информацию по графу состояний, и на экране должен быть отображен граф и собранная по нему статистика. Однако пользователю может потребоваться дополнительная информация по любой из отображенных вершин. Исходя из этого, было принято:

* при вызове интерфейса выводить на экран только граф и статистику по нему;
* при необходимости отображать информацию по интересующей вершине по клику мыши.

Для удобства использования было также решено именовать окна интерфейса согласно тем точкам принятия решений, для которых они были вызваны.

4.2. Отображение статистики

Интерфейс подсистемы реализует отображение на экране статистики двух типов:

* статистика по результатам поиска на графе пространства состояний;
* статистика по выделенной вершине графа, т. е. статистики для отдельного состояния.

Для составления перечня статистических данных, которые необходимо отобразить в интерфейсе разрабатываемой подсистемы, необходимо учесть, что в программном комплексе RAO‑XT уже существует модуль трассировки, отображающий различную информацию о работе системы и результатах моделирования. Назначение разрабатываемой подсистемы – устранить недостаток текстового формата вывода трассировщика, т. е. визуализировать поиск решения на графе пространства состояний с отображением его древовидной структуры, и не потерять при этом в информативности.

4.2.1 Вывод статистики по результатам поиска

Всю статистику по работе модели, включая данные, относящиеся к алгоритму поиска, пользователь может получить из окна трассировщика, поэтому во избежание лишнего дублирования информации принято решение в разрабатываемом интерфейсе отображать только ту статистику, которая относится непосредственно к результатам поиска:

* стоимость решения – фактическая стоимость найденного пути от исходной вершины до целевой;
* количество раскрытых вершин – количество вершин, для которых были порождены потомки;
* количество вершин в графе – количество вершин, добавленных в граф в процессе поиска.

Перечисленный список является необходимым минимумом, который должен быть представлен пользователю разрабатываемой подсистемой.

4.2.2. Вывод статистики по вершине

Необходимая информация по результатам поиска представлена совокупностью информации, предоставляемой графом пространства состояний, и статистикой по нему. Если же пользователю требуется дополнительная информация по какой-либо вершине, он сможет получить ее, выделив ее кликом. Это сделано для того, чтобы исключить из действий пользователя операцию обратного переключения к интерфейсу модуля трассировщика, что позволит повысить эффективность работы и сэкономить время.

Модуль трассировки предоставляет следующую информацию при порождении новых вершин:

* номер самой вершины;
* номер ее родителя;
* значение фактической стоимости пути от исходной вершины до текущей;
* значение оценочной стоимости пути от текущей вершины до целевой;
* имя использованной активности с указанием в скобках имен релевантных ресурсов;
* значение стоимости применения правила.

Все перечисленные данные необходимо вывести на экран пользователя.

4.3. Вызов окна интерфейса подсистемы из интерфейса модуля трассировки

На этапе формирования ТЗ в требованиях к функционалу подсистемы было указано, что вызов ее интерфейса должен осуществляться из интерфейса модуля трассировки. Это требование предъявлено к подсистеме по следующим причинам:

* модуль трассировки является основным средством вывода информации о состоянии и работе системы RAO-XT;
* в общем случае модель может содержать несколько точек принятия решений типа search.

Ситуация, когда при наличии в системе точек принятия решений типа search, статистика по ним не выведена в полотно трассировки, может возникнуть только в случае, когда пользователь вручную отключил вывод трассировки точек принятия решений, и, следовательно, не интересуется этой информацией. В любой другой ситуации пользователь будет иметь перед глазами записи, относящиеся к поиску на графе пространства состояний. К ним целесообразнее всего привязать вызов разрабатываемого модуля.

Ситуация с несколькими точкам принятия решений в модели так же является веским доводом против других вариантов вызова окна интерфейса (например, отдельной кнопки в меню), т.к. возникнет очевидная неоднозначность, граф какой точки хочет увидеть пользователь. Вызов окна интерфейса с привязкой к строке трассировщика исключит такую ситуацию, потому что каждая запись в полотне трассировки относится к соответствующей ей точке принятия решений.

4.4. Модуль визуализации в системе RAO-XT

На этапе концептуального проектирования ключевой задачей является разработка правильной схемы взаимодействия подсистемы визуализации с остальными частями системы RAO-XT. Ниже приведены основные требования к модулю трассировки как к компоненту системы:

* Сохранение информации о состоянии системы в каждый момент модельного времени производится компонентом *Database* в бинарном формате. Таким образом компонент *Database* осуществляет сериализацию данных, т.е. сохранение их в компактном, но недоступном для восприятия пользователем формате. Модуль визуализации должен уметь интерпретировать эти данные и формировать на выходе древовидную структуру, по которой можно будет восстановить и построить граф пространства состояний и затем вывести его на экран пользователя.
* Модуль трассировки должен предоставлять выходные данные исключительно для пользователя. Никакие другие компоненты системы не должны основывать свою работу на выходных данных модуля визуализации.
* Модуль трассировки должен состоять из двух частей:
  + библиотечная часть;
  + графическая часть;
* Библиотечная часть модуля вывода трассировки должна заниматься исключительно преобразованием бинарных данных в древовидную структуру. Далее с полученной структурой работает графическая часть модуля, расположенная в пакете *UI*. Эта часть отвечает за отрисовку графа на экране и за графический интерфейс пользователя. При этом данные, которые отображает графическая часть модуля, должны полностью соответствовать тем данным, которые на данный момент преобразовала библиотечная часть.
* Разрабатываемый модуль должен быть активен только после прогона модели. Модуль выводит результат на экран, основываясь на данных, сформированных только после окончания прогона модели.

5. Технический этап проектирования

5.1. Проектирование библиотечной части подсистемы визуализации

5.1.1. Формирование древовидной структуры

Библиотечная часть подсистемы визуализации, основным компонентом которой является класс *TreeBuilder*, должна взаимодействовать непосредственно с содержимым базы данных и формировать на его основе древовидную структуру графа. База содержит в себе записи с сериализованными данными о результатах работы системы, результатах моделирования и поиска. Типов записей в базе данных пять:

* Системные. Это записи, содержащие информацию о произошедших системных событиях;
* Записи ресурсов. Это записи, содержащие информацию об изменении состояния ресурсов модели;
* Записи образцов. Это записи, содержащие информацию о выполнившихся активностях и событиях;
* Записи поиска по графу. Это записи, содержащие информацию о порядке выполнения поиска по графу и найденном решении;
* Записи результатов. Это записи, содержащие информацию об изменении текущего значения результата.

Для разрабатываемой подсистемы интерес представляет только один тип записи, относящийся к поиску решения на графе. Сама запись поиска также бывает нескольких типов:

* начала поиска;
* раскрытия вершины;
* порождения новой вершины;
* решения;
* завершения поиска.

Зная правила сериализации данных для записи каждого типа, модуль должен считать нужное количество байт из записи, преобразовать их в переменную корректного типа и записать эту переменную в структуру. Для описания вершины графа описан вложенный по отношению к основному классу *TreeBuilder* класс *Node*. Экземпляр такого класса должен полностью охарактеризовывать конкретную вершину графа.

После чтения всех записей поиска, находящихся в базе данных, классом *TreeBuilder* будет сформирована полная структура дерева графа.

5.1.2. Чтение записи начала поиска

Встреча записи начала поиска в базе данных означает, что система осуществила поиск для определенной точки принятия решений типа search, описанной в тексте модели. В соответствии с этим библиотечная часть модуля визуализации должна отреагировать действием начала формирования дерева нового графа. Для этого необходимо:

* считать из записи номер соответствующей точки принятия решений;
* создать экземпляр класса *Node* с параметрами, соответствующими корневой вершине дерева, и добавить его в структуру дерева графа.

Корневая вершина обладает нулевым индексом (нумерация вершин в системе RAO-XT начинается с нуля, в отличие от системы RAO-Studio, где она начиналась с единицы) и не имеет предка.

5.1.3. Чтение записи раскрытия вершины

Запись раскрытия вершины содержит информацию о номере вершины, для которой будут порождены потомки, на определенном шаге поиска, и эта информация никак не интерпретируется разрабатываемым модулем. Записи данного типа им рассматриваться не должны.

5.1.4. Чтение записи порождения новой вершины

Запись порождения новой вершины бывает трех типов:

* порождение новой вершины. Вершина с таким состоянием системы не содержится в уже построенной части графа;
* порождение лучшей вершины. Вершина с таким состоянием уже есть в графе, и она перезаписывается, поскольку вновь найденный путь имеет меньшую стоимость;
* порождение худшей вершины. Вершина с таким состоянием уже есть и она не включается в граф (вновь найденный путь имеет большую стоимость).

Вершины, описываемые первыми двумя типами записей, добавляются на граф состояний. Обновление стоимостей при нахождении лучей вершины происходит внутри класса *DecisionPointSearch* библиотеки RAO-XT, поэтому внутри записей уже содержится вся необходимая информация, и операции, которые должен выполнить модуль при встрече записей этих типов, не отличаются. Модуль визуализации должен выполнить для них следующие действия:

* считать номер вершины и ее предка;
* считать стоимости *g* и *h*;
* определить имя активности;
* считать стоимость применения правила;
* создать экземпляр класса *Node* и добавить его в структуру графа.

Следует учитывать, что вершина имеет отношение к графу определенной точки принятия решения, номер которой модуль определил, встретив ранее запись начала поиска.

Вершина, описываемая последним типом записи порождения вершины, не добавляется на граф, а значит запись данного типа не должна рассматриваться разрабатываемым модулем.

5.1.5. Чтение записи решения

Записи данного типа присутствуют в полотне трассировке в случае существования решения и содержат информацию о том, какие правила были применены к исходной вершине графа (исходному состоянию системы) для перехода в целевое состояние. Так же они содержит номера вершин, входящих в решение, что и будет использоваться модулем визуализации.

Прочитав записи данного типа, класс *TreeBuilder* заполнит список, содержащий вершины, принадлежащие решению для текущей точки принятия решения.

5.1.6. Чтение записи завершения поиска

Среди записей данного типа модулем визуализации будут использоваться записи, обладающие следующими признаками:

* успешное завершение поиска. Наличие в базе данных записи такого типа означает, что решение найдено;
* неудачное завершение поиска. Наличие такой записи означает, что решение отсутствует.

Записи, обладающие признаками, отличными от этих, не рассматриваются разрабатываемым модулем, так как построение графа пространства состояний возможно только для перечисленных типов записей.

Запись завершения поиска содержит в себе информацию о результатах поиска на графе для текущей точки принятия решений, следовательно необходимо описание структуры, в которую она будет считана, и которая будет использоваться в дальнейшем для вывода статистики. Для описания такой структуры был описан вложенный по отношению к библиотечному классу *TreeBuilder* класс *GraphInfo*. Класс должен содержать поля для записи в них информации, описанной в п. 4.2.1.

В случае чтения записи об успешном завершении поиска, из нее необходимо считать данные по всем полям класса *GraphInfo*.

В случае чтения записи о неудачном завершении поиска, данные считываются только по двум полям:

* количество открытых вершин;
* количество вершин в графе.

Стоимость решения в таком случае необходимо задать равной нулю.

Данные из записи считываются модулем для текущей точки принятия решений.

5.2. Проектирование графической части подсистемы визуализации

5.2.1. Выбор графической библиотеки