электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 519.85

Управление конфигурацией изделия, с использованием метода «Программирование в ограничениях»

 # 09, сентябрь 2012

Коновалов Н.А.

Студент,
кафедра «Компьютерные системы автоматизации производства»

Научный руководитель: Овсянников М.В.,
к. т. н., доцент, заместитель заведующего по методической работе кафедрой «Компьютерные системы автоматизации производства»

МГТУ им. Н.Э. Баумана
rk.konovalov@mail.ru

Деятельность многих современных предприятий тесно связанна с PDM-системами. PDM-система - это система управления данными об изделии, которая позволяет решить две проблемы, возникающие при разработке и поддержке жизненного цикла продукции: управление данными об изделии и управление информационными процессами жизненного цикла изделия, создающими и использующими эти данные. Правильное управление информацией об изделии является основополагающим фактором, обеспечивающим выпуск конкурентно-способной продукции. В рамках PDM-систем существует такое направление, как управление конфигурацией изделия. Одной из основных задач управления конфигурацией является синтез конфигурации по требованиям заказчика.

При разработке систем синтеза конфигурации обычно используются стандартные методы и алгоритмы, что приводит к необходимости разработки новых алгоритмов при малейшем изменении поставленной задачи. Подобное решение приводит к потерям времени и дополнительным затратам на доработку системы. Альтернативным способом решения данной задачи является метод удовлетворения ограничений (Constraint Satisfaction Problem).

Программирование ограничениями за последние пару десятков лет стало востребованным для решения задач прикладного характера, целью которых является получение наиболее оптимального решения с заданными характеристиками. Об этом говорит и покупка крупной компанией IBM среды ILOG Optimizer в 2009 году, и вовлечении еще одного IT-гиганта Google в создание собственного решателя CP-задач, а так же появления десятков других решателей, создающихся как независимыми программистами, так и более мелкими компаниями.

Востребованность возникает из-за того, что для решения задач подобного типа не нужно каждый раз заново писать полностью алгоритм поиска решений – достаточно задействовать функции решателей, что позволяет создавать более гибкие решения в более короткие сроки. Так же это позволяет сократить штат

разработчиков, потому что их усилия теперь направляются только на описание задачи и задание ограничений в выбранном решателе. Сокращение сроков и штата разработчиков, гибкость решения задачи, приводят к непосредственному снижению затрачиваемых средств, что является одним из важных достоинств в условиях конкурентного рынка.

Задача и модель представляются как совокупность отношений, которые соответствуют связям, существующим между переменными задачи. Эти отношения, называемые общим термином "ограничения", могут иметь вид уравнений, неравенств, логических выражений и т.п. Метод программирования в ограничениях применяется для поиска конфигураций, наиболее точно удовлетворяющих требованиям.

В классической постановке проблема состоит из нескольких элементов:

- Набор переменных $X = \{x_1, x_2, : x_n\}$;
- Для каждой переменной x_i задан ограниченный домен D_i тех значений, которые она может принимать;
- Задан набор constraint, ограничивающих значения, которые данные переменные могут принимать;

В общей постановке задачи значения переменных не обязательно целые, они даже не обязаны быть числовыми. Решение задачи – это набор значений, присвоенных переменным таким образом, что все ограничения выполняются. Способов поиска решения огромное множество: от систематических до стохастических.

Под ограничениями понимаются аксиомы, которые могут, как ограничивать значение отдельного свойства объекта конфигурации, так и связывать произвольное число свойств различных объектов конфигурации. Ограничения содержаться в функциональной конфигурации и являются основой онтологии предметной области задачи. Ограничения имеют форму логических или функциональных выражений, а также таблиц значений. Функциональные выражения могут быть использованы решателем для вычисления любого аргумента при остальных известных. Если какие либо из аргументов заданы в виде значения сложной формы (диапазоны, списки значений и т.д.) то искомое значение параметра так же будет в сложной форме.

Все виды ограничений задачи существуют в виде единого пространства аксиом и разрешаются (вычисляются) или проверяются решателем как единая система выражений. Разрешение ограничений происходит итерационным обходом пространства аксиом с вычислением промежуточных значений параметров, пока не будут найдены и проверены все параметры, перечисленные в требованиях, или не будет зафиксирован факт неполноты или невыполнимости ограничений для проверяемой конфигурации.

Аксиомы необходимы при синтезе конфигурации для доопределения значений некоторых свойств на этапе наполнения структуры, а также при проверке конфигурации на соответствие требованиям. Использование аксиом позволяет обуславливать применение одних компонентов в конфигурации значениями свойств других.

Каждая такая аксиома может быть реализована в виде n-сторонней функциональной зависимости, т.е. в виде одной и той же зависимости, но выраженной через разные аргументы.

Рассмотри распространенные методы структурного синтеза:

- Метод ветвей и границ. (Группа методов дискретной оптимизации, основанных на разделении множества альтернатив на подмножества и отсечении неперспективных подмножеств.)
- Метод поиска с запретами tabu search. (Метод локального поиска с запретами на повторное исследование точек, пройденных на нескольких последних шагах поисковой оптимизации.)
- Метод распространения ограничений constraints propagation. (Метод оптимизации, сводящийся к сужению допустимых интервалов управляемых переменных с помощью учета (распространения) исходных ограничений на выходные параметры.)
- Генетический алгоритм. (Приближенный метод оптимизации, использующий некоторые из принципов приспособления живых организмов к окружающей среде.)
- Метод отжига. (Метод минимизации целевой функции, имитирующий минимизацию потенциальной энергии тела в процессе отжига деталей. В методе отжига допускается переход с некоторой вероятностью в точки с худшим значением целевой функции ради увеличения возможностей выхода из локальных экстремумов.)
- Мозговая атака. (Метод синтеза проектных решений, относящийся к неформальным методам решения задач изобретательского характера.)
- Дерево построения. (Последовательность операций разработки твердотельной модели, упорядоченная по времени совершения операций, т.е. история создания модели.)
- Синхронное моделирование. Метод синтеза и редактирования твердотельных моделей, предложенный компанией Siemens PLM Software и основанный на автоматическом определении при моделировании имеющихся ограничений и условий сопряжения деталей в сложных узлах благодаря применению экспертных систем.

Примеры задач структурного синтеза:

- задача JSSP (Job Shop Scheduling Problem). Одна из задач синтеза расписаний;
- маршрутизация транспортных средств. Задача выполнения заказов на доставку продуктов из одних пунктов в другие пункты, решаемая методами дискретной оптимизации. В общем случае заданы временные окна на выполнение заказов, а определению подлежат пункты-источники продуктов и транспортные средства для каждого заказа
- задача VRPTW (Vehicle Routing Problem with Time Windows) Задача маршрутизации транспортных средств с временными окнами. Синтез сети каналов передачи данных. Задача выбора трасс для прокладки линий связи и распределения трафика по линиям, решаемая методами дискретной оптимизации

Существует множество решателей, позволяющие описывать модели управления данными об изделии, причем решатель может поставляться в виде

законченного программного продукта или библиотеки функций. Он может быть написан на одном из распространенных языков программирования (C++, Java, PHP, LISP, Prolog и т д.). Для исследования функций и возможностей метода «программирование в ограничениях» была выбрана среда ILOG Optimizer, компании IBM, входящая в состав IBM ILOG CPLEX Optimization Studio, и является решающей программой на основе программирования в ограничениях, которая обеспечивает решение модельных и других задач с использованием алгоритмов распространения и поиска ограничений.

В качестве исследуемой задачи, была выбрана задача построения школьного расписания. Сформулирована она таким образом, что бы она удовлетворяла требованиям задач управления конфигураций изделий. В нашем случае будет рассматриваться проблема контейнерной задачи, в которой требуется разместить N элементов в L ячеек по определенным условиям (или иными словами ограничениям).

Исходными данными являются данные о преподавателях: Фамилия и Предмет, который ведет преподаватель; данные о занятиях: перечислены доступные классы, и указано какой предмет поддерживает каждый из классов; Данные о группах: для каждой группы указано количество предметов на неделе.

Запишем начальные ограничения с помощью математической записи:

- Аудитория должна быть пригодна для предмета:

$$\mathrm{Support}_{\mathrm{pr,aud}} = \begin{cases} 1, \text{Если предмет pr может проводится в аудитрории aud} \\ 0, \text{Если не может} \end{cases}.$$

В виде матрицы соответствия, это ограничение можно представить в следующем виде, где закрашенные ячейки обозначают 1, а пустые 0 (табл. 1).

Таблица 1

Матрица соответствий первому ограничению

		Аудитории							
		R1	R2	R3	R4	R5	R6	Стадион	Лабор.
Предмет	Математика	■	■	■	■	■	■		
	Французкий	■	■	■	■	■	■		
	Экономика	■	■	■	■	■	■		
	Физика	■	■	■	■	■	■		
	Биология								■
	Физкультура							■	
	История	■	■	■	■	■	■		
	География	■	■	■	■	■	■		
	История	■	■	■	■	■	■		

- Некоторые дисциплины не могут стоять рядом друг с другом:

$$\mathrm{S}_{\mathrm{pr1,pr2}} = \begin{cases} 1, \text{Если предмет Pr1 не может стоять после} \\ \quad \text{или перед предметом pr2} \\ 0, \text{Если может} \end{cases}.$$

Матрица соответствия для этого ограничения, где также закрашенные ячейки обозначают 1, а пустые 0 (табл. 2).

Таблица 2

Матрица соответствий второму ограничению

Предмет		Математи	Французски	Экономика	Физика	Биология	Физкульту	История	География	История
	Математика									
	Французский									
	Экономика									
	Физика									
	Биология									
	Физкультура									
	История									
	География									
	История									

Запишем накладываемые ограничения:

- Преподаватель не может одновременно находиться в разных аудиториях. В аудитории не может быть больше двух преподавателей. Те же ограничения накладываются на группу, которой преподается предмет.

$$S_{prep,aud,gr,t} = \begin{cases} 1, \text{Если преподаватель prep преподает в аудитории aud,} \\ \quad \text{в группе gr, в момент времени t} \\ 0, \text{Если нет, и если } pr1 = pr2 \end{cases},$$

$$\sum_{aud} \sum_{prep} S_{aud,prep} \leq 1,$$

$$\sum_{aud} \sum_{gr} S_{aud,gr} \leq 1.$$

Матрица соответствия в этом случае будет представлять многомерный куб, ребрами которого будут перечисленные переменные. Абсолютно все ячейки также будут заполнены нулями и единицами.

Должны быть известны наименования и количество групп, названия предметов, длительность каждого занятия и число занятий для каждого предмета, а так же доступные аудитории:

gr_i = «string»; pr_i = «string»; dur_i = a; $povt_i$=b; aud_i= «string»;

Далее все описание задачи ведется на языке C++, с использованием функций, предоставляемой средой ILOG, позволяющие решить данную задачу.

Первым шагом необходимо задать исходные данные. Данные могут быть объявлены в коде самой программы в разделе данных, в качестве переменных или многомерных массивов на языке C++. Либо находиться во внешнем файле, базы данных Access или Excel. Массивы, содержащие различные данные по типу, должны определяться через типы.

Дальнейшим важным шагом является написание так называемых «определенных массивов», в которых не указываются никаких данных, а лишь объявляются возможные принимаемые значения и диапазоны. Запись ведется с помощью функций среды ILOG. В эти массивы в конечном результате функции вернут наши искомые данные. В нашем случае это массив из различных типов данных, представляющий собой готовое расписание на неделю.

Далее записываются накладываемые ограничения на весь массив или на отдельные переменные. Форма записи ведется с помощью обычных математических и логических операторов:

$$переменная\_N \geq \frac{переменные\_столбца\_a - переменная\_c}{переменные\_столбца\_b}.$$

После запуска программы будут найдены все возможные доступные комбинации, и заполнены определенные массивы в соответствие с ограничениями одной из найденных комбинаций, которые можно будет вывести во внешний файл или в log программы, а так же вывести все связанные с ними данные. В нашем примере это будет массив типов, с готовым расписанием на неделю. Название предметов и имена преподавателей приведены латинскими буквами (табл. 3).

Таблица 3 Результат работы алгоритма

		группа X			группа Y			группа Z		
		Ауд.	Предм.	Препод.	Ауд.	Предм.	Препод.	Ауд.	Предм.	Препод.
ПН	1	R5	Maths	Jean-Francois				R4	French	Pierre
	2	R5	Maths	Jean-Francois				R4	French	Pierre
	3				R6	French	Pierre			
	4				R4	Physics	Paul			
	5	R5	French	Pierre				R5	Maths	Jean-Francois
	6	R2	Physics	Paul						
ВТ	1									
	2				R1	French	Pierre			
	3									
	4	R4	Maths	Jean-Francois				R2	French	Pierre
	5	R4	Maths	Jean-Francois				R2	French	Pierre
	6	R3	French	Pierre	R4	Maths	Jean-Francois	R2	Physics	Paul
СР	1									
	2	R6	Maths	Jean-Francois				R4	French	Pierre
	3	R6	Maths	Jean-Francois				R4	French	Pierre
	4				R4	French	Pierre	R1	Maths	Jean-Francois
	5	R1	Physics	Paul	R3	Maths	Jean-Francois			
	6	R1	Physics	Paul						
ЧТ	1	R1	Maths	Jean-Francois				R3	French	Pierre
	2	R1	Maths	Jean-Francois				R3	French	Pierre
	3									
	4									
	5									
	6									

ILOG позволяет найти требуемые данные различными алгоритмами поиска:

- Depth first (поиск в глубину)
- Reset (сброс)
- Multipoint (комбинированный)

Для каждого метода можно настроить ряд параметров, отвечающих за поиск. Одним из таких параметров является количество неудачных раскрытых вершин, для сброса раскрытия этой ветки. Для конкретной задачи наиболее оптимален метод поиска Depth first, так как он показал не только минимальное время работы, но и наибольшее количество раскрытых вершин.

В заключение, стоит сказать, что метод «программирование в ограничениях» получает все большее и большее распространение. Связано это с широким кругом задач, которые он способен решать. Ранее это были лишь библиотеки функций, написанные энтузиастами-разработчиками. Сейчас выпуском таких продуктов и решений занимаются компании с мировым именем. На данный момент список таких решений измеряется десятками. Одной из них была рассмотренная среда ILOG компании IBM.

Но у этих систем есть и свои недостатки. К каждой поставленной задаче требуется свой индивидуальный подход. Необходимо продумать каждую деталь, основываясь на опыте и знании тонких мест среды, в которой идет описание.

Объявление и переобъявление типов данных и определенных массивов бывает крайне запутанным и непонятным, дабы достичь поставленных целей. Не говоря уже об оптимизации модели, в которой необходимо настроить много параметров, для подходящего алгоритма поиска. Но это лишь говорит о широких возможностях данного метода.

Литература

1. Ibm ilogo pllanguageuser's manual, 2009.- 313 с.
2. Внутренний справочник IBM ILOC CP OPTIMIZER.