Структуры хранения данных в программной системе параллельного поиска глобально-оптимальных решений Global Expert

А.О. Казаков, А.В. Сысоев

Нижегородский государственный университет им. Н. И. Лобачевского

Данная работа посвящена способам организации поисковой информации, накапливаемой в процессе поиска глобально-оптимального решения, в системе Global Expert [1], а также сравнению их эффективности.

Постановка задачи

Задачи глобальной многоэкстремальной оптимизации широко встречаются во многих областях науки и техники. Аналитически такие задачи могут быть решены лишь в весьма простых случаях, поэтому обычным способом поиска глобально-оптимального решения являются различные численные итерационные процедуры. В системе Global Expert используется один из эффективных методов глобального поиска – индексный метод [2], накапливающий и использующий в ходе решения поисковую информацию, объём которой непрерывно растёт. Таким образом, наличие эффективной структуры хранения поисковой информации, обеспечивающей быстрый доступ и обработку, является необходимым условием работы поисковых методов.

Понятие матрицы состояния поиска (МСП)

В процессе оптимизации запоминается информация обо всех точках испытаний. Для каждой точки она включает:

- x – координату точки испытания;
- z – значение функции в точке x;
- d – расстояние до следующей точки;
- ix – номер итерации;
- iz – индекс точки;
- R – характеристику интервала.

Набор перечисленных параметров будем именовать *информационным элементом* или *столбцом* поисковых данных.

Совокупность информационных элементов, упорядоченную по координате, будем называть *матрицей состояния поиска* (МСП).

Формы организации МСП

На данный момент в системе Global Expert реализованы 3 базовые структуры хранения МСП – таблицы, деревья и АВЛ-деревья, а также структуры в виде их блочных представлений. Для обеспечения быстрого поиска интервала с максимальной характеристикой в системе используется приоритетная очередь характеристик совместно с вышеуказанными структурами.

Базовые формы организации МСП

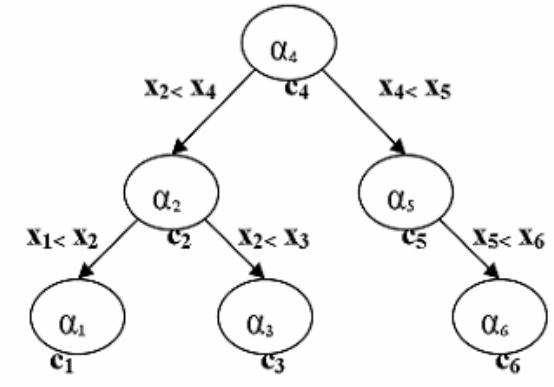
В основе структур организации МСП лежит выделение непрерывного куска памяти под хранение поисковой информации. Выделенная память состоит из двух частей: первая заполнена уже имеющимися данными, представление в памяти которых зависит от типа структуры; вторая - список свободных звеньев.

Динамические Таблицы

Данная структура реализована в виде линейного списка. На каждом шаге работы поискового метода появляется одна или несколько новых точек, которые необходимо добавить в МСП с сохранением упорядоченности. При добавлении элемента в матрицу, он размещается в начале свободной области и упорядоченно по координате *x* вставляется в список. Операции вставки и поиска имеют линейную от числа элементов трудоёмкость.

Деревья

В данном случае матрица организуется по принципу бинарных поисковых деревьев. Дерево строится по правилу поддержания упорядоченности элементов матрицы по координате *x*. Операции вставки и поиска по *x* будут иметь трудоёмкость O(h), где h – высота дерева. Известно, что средняя высота случайного двоичного дерева равна O(logN), однако в худшем случае может достигать O(N). Недостаток данной структуры состоит в том, что основные операции при работе с матрицей связаны с поиском элементов по их порядковым номерам, а реализовать их можно только через обход дерева, сложность которого составляет O(Nh). Для устранения данного недостатка используется следующий подход: для каждой вершины дерева накапливается вес (c_i) - число элементов в левом поддереве плюс 1. Зная веса элементов, можно восстанавливать их порядковые номера. За счёт использования данного подхода, поиск элементов в дереве по их порядковому номеру также будет иметь трудоёмкость O(h). При этом после вставки нового элемента нет необходимости пересчитывать веса для всей матрицы, а достаточно только для той ветви, по которой будет происходить спуск.



АВЛ-деревья

Основные операции с двоичным поисковым деревом высоты h могут быть выполнены за O(h) действий, но, если не принимать специальных мер при выполнении операций, малая высота не гарантируется. Для повышения эффективности операций используют различные приёмы балансировки деревьев. Одним из видов сбалансированных деревьев поиска являются АВЛ-деревья (Адельсон-Вельский Г.М. и Ландис Е.А.), гарантирующие оценку высоты величиной O(logN).

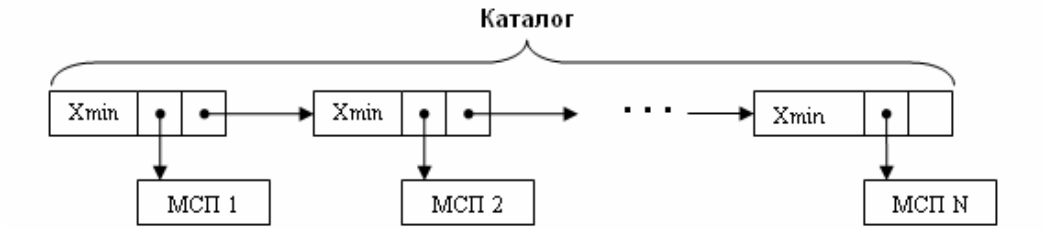
Двоичное поисковое дерево является АВЛ-сбалансированным (АВЛ-деревом) тогда и только тогда, когда для каждого его узла высоты его левого и правого поддеревьев отличаются не более чем на единицу. Если для какого-то узла это условие перестало выполняться, то за счёт проведения локальной операции поворота сбалансированность восстанавливается во всём дереве. Следует отметить, что сбалансированность может нарушить только операция вставки в дерево, остальные операции выполняются так же как в обычном бинарном дереве поиска.

Представление матрицы в виде набора блоков

Общим недостатком описанных выше структур является необходимость наличия непрерывного куска памяти для хранения поисковой информации. Объём памяти требуемый для работы системы пропорционально зависит от количества итераций. При большом объёме вычислений и сильной фрагментации увеличивается вероятность того, что необходимого куска непрерывной памяти найдено не будет.

Для обеспечения более быстрого доступа к элементам МСП, а также для устранения описанного выше недостатка разработан блочный способ представления матрицы. Блоком назовём непрерывный фрагмент МСП, который сам является МСП. При этом каждый столбец МСП должен находиться ровно в одном блоке, а порядок следования блоков определяется порядком следования столбцов в МСП. Блоки матрицы связаны между собой через каталог.

Представление каталога блоков в виде линейного списка

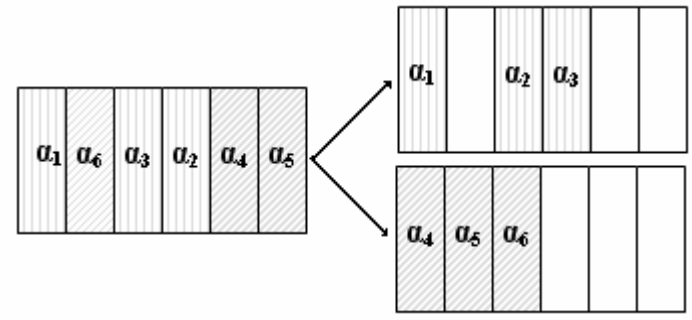


Блоки матрицы связаны между собой через каталог, имеющий вид линейного списка. При этом каждому блоку в каталоге соответствует заголовок, содержащий следующую информацию:

- *Xmin* – минимальная координата *x* среди всех столбцов блока;
- указатель на блок, соответствующий данному заголовку;
- указатель на следующий заголовок.

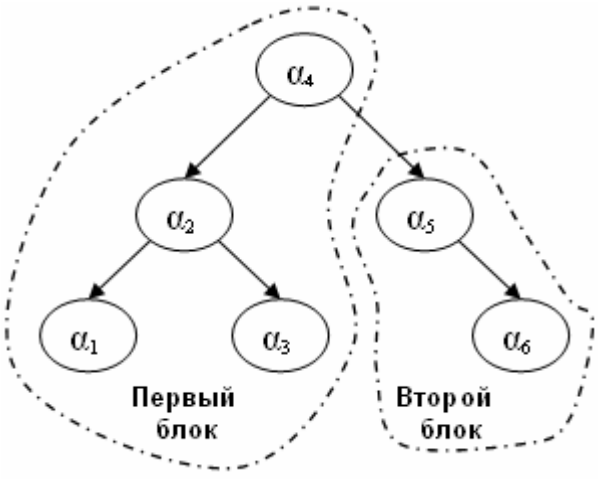
Заголовки упорядочены по значению *Xmin*. Таким образом, перемещаясь по каталогу (вместо прохода по всей матрице), нужно найти необходимый блок, после чего вести работу только с ним. Как только размер блока становится больше некоторого заданного, происходит его деление пополам. Для нового блока в каталоге создается заголовок. Описание деления на блоки для каждого типа матриц указано ниже.

Деление блока (таблицы)



При попытке вставить элемент в блок динамической таблицы, свободная область в котором закончилась, создается новый блок и половина элементов (с большими значениями по координате *x*) из старого блока копируется в новый блок. В каталоге создается заголовок, отвечающий новому блоку.

Деление блока (деревья, АВЛ-деревья)

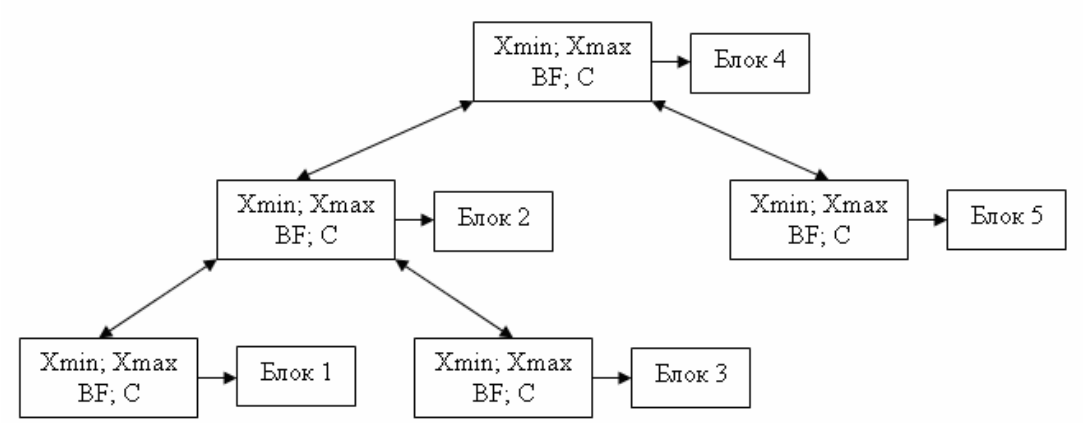


В правом поддереве находятся элементы с большими координатами *x*, чем у элементов в левом. На основе правого поддерева можно построить новое дерево (новый блок). Плюсом такого подхода является то, что веса, соответствующие элементам, при делении не меняются, следовательно, не придется делать пересчет. Но, следует отметить, что число элементов в левом и правом поддереве может сильно отличаться., поэтому перед делением блока дерево необходимо сбалансировать.

Представление каталога блоков в виде АВЛ-дерева

Каталог в виде линейного списка имеет существенный недостаток: поиск необходимого в нём блока производится с линейной трудоёмкостью O(n), где n – длина ка-

талога (количество блоков). Далее будет рассмотрено представление каталога в виде АВЛ-сбалансированного дерева (АВЛ-каталог).



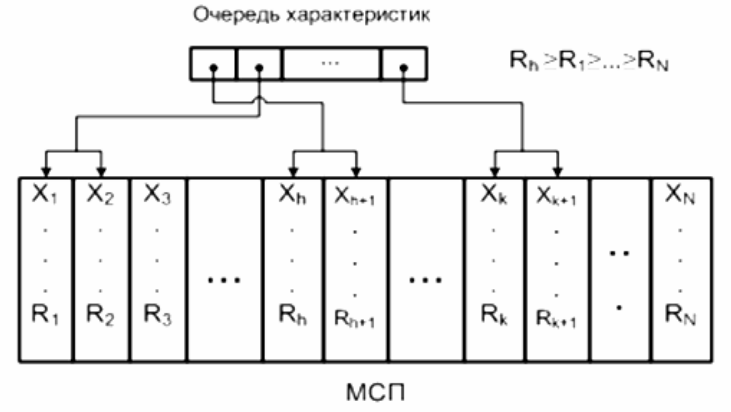
Блоки матрицы связаны между собой через каталог, имеющий вид АВЛ-дерева. При этом каждому блоку в каталоге соответствует заголовок, содержащий следующую информацию:

- *Xmin* – минимальная координата *x* среди всех столбцов блока;
- *Xmax* – максимальная координата *x* среди всех столбцов блока;
- *BF (BalancingFlag)* – показатель сбалансированности блока. Необходим для поддержания АВЛ-свойства;
- C – число элементов в левом поддереве. Необходимо для поиска столбца по порядковому номеру;
- указатель на блок соответствующий данному заголовку;
- указатель на левый заголовок;
- указатель на правый заголовок;
- указатель на родительский заголовок. Используется при поиске блока, нарушающего балансировку.

Заголовки упорядочены по значению *x* следующим образом: каждый блок содержит столбцы принадлежащие диапазону [Xmin, Xmax], столбцы с координатами меньшими чем Xmin содержатся в заголовке из левого поддерева, а столбцы, у которых координата *x* больше Xmax содержатся в заголовке из правого поддерева. Таким образом, перемещаясь по каталогу (вместо прохода по всей матрице), нужно найти необходимый блок, после чего вести работу только с ним. Как только размер блока становится больше максимально допустимого, происходит его деление пополам (согласно описанным выше алгоритмам). Поиск в таком каталоге осуществляется с логарифмической трудоёмкостью.

Очередь характеристик

Служит для быстрого поиска интервала с максимальной характеристикой (без прохода по всей МСП). В неё помещаются интервалы из МСП, характеристики которых больше других.



Максимальное число интервалов хранящихся в очереди ограничивается ее размером. Каждый элемент очереди содержит указатели на левую и правую границу интервала. Причем очередь является приоритетной: интервалы хранятся упорядо-

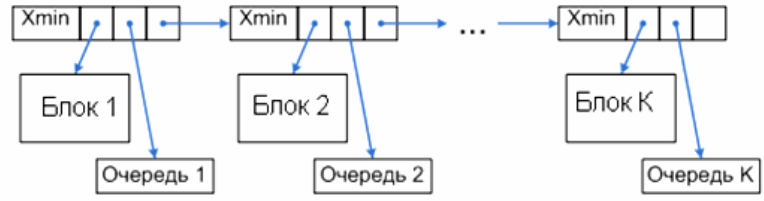
ченно по убыванию своих характеристик. Наличие очереди позволяет определить интервал очередной итерации поиска без прохода по МСП.

Использование одной очереди характеристик для МСП имеет ряд недостатков.

- Очередь имеет постоянный максимальный размер; однако, с ростом числа элементов в МСП имеет смысл увеличивать очередь;
- При использовании очередей и МСП с блочным представлением необходимо перезаполнять очередь, если в матрице произошло деление блока, а эта процедура затратная.

Блочное представление МСП с локальными очередями характеристик

Идея данного подхода в том, что вместо одной глобальной очереди для всей МСП для каждого блока организуется своя локальная очередь.



При этом может опустеть только локальная очередь и заполняться она будет только по своему блоку. При делении блока перезаполняются очереди только для 2 блоков, а не для всей матрицы. С использованием локальных очередей число элементов с лучшими характеристиками, хранящихся в очередях, растет вместе с ростом объема поисковой информации.

Следует отметить, что на данный момент в системе GlobalExpert реализованы локальные очереди только для блочных структур с линейным каталогом.

Результаты

По результатам проведённых экспериментов с использованием рассмотренных структур установлено, что среди базовых структур наиболее эффективными оказались деревья; среди блочных структур самыми быстрыми являются блочные таблицы с каталогом в форме АВЛ-дерева, а блочные деревья с каталогом в форме АВЛ-дерева оказались эффективнее блочных деревьев с линейным каталогом. Для всех блочных структур были подобраны следующие оптимальные размеры блока:

- для блочных таблиц с линейным каталогом 2000;
- для блочных деревьев с линейным каталогом 4000-6000;
- для блочных таблиц с каталогом в форме АВЛ-дерева 500-2000;
- для блочных таблиц с каталогом в форме АВЛ-дерева 4000-6000.

Среди всех структур самыми быстрыми оказались блочные деревья с локальными очередями. За счёт внедрения локальных очередей трудоёмкость самой тяжелой операции (поиск интервала с максимальной характеристикой) удалось уменьшить для блочных таблиц в 16 раз, а для блочных деревьев в 57. Рекомендуемый размер очереди находится в диапазоне от 20 до 100.

Следует отметить, что для блочных деревьев с каталогом в форме АВЛ-дерева трудоёмкость операций (кроме поиска интервала с максимальной характеристикой) существенно меньше соответствующих операций для остальных блочных структур. Поэтому за счёт реализация для них очередей можно получить более эффективную структуру, чем блочные деревья с локальными очередями.

Работа выполнена при поддержке Совета по грантам Президента Российской Федерации (грант № МК-1536.2009.9) и Федерального агентства по науке и инновациям (ФЦП «Научные и научно-педагогические кадры инновационной России», госконтракт № 02.740.11.5018

Литература

1. Гергель В.П, Сысоев А.В. О реализации параллельной версии индексного метода поиска глобально-оптимальных решений в многомерных задачах оптимизации в программном комплексе «Абсолют Эксперт» // Труды всероссийской конференции «Научный сервис в сети Интернет: технологии параллельного программироания». 2006. С. 115-118.
2. Strongin R.G., Sergeev Ya.D. Global optimization with non-convex constraints: Sequential and parallel algorithms. Kluwer Academic Publisher, Dordrecht, 2000.
3. Алексеев В.Е., Таланов В.А. Графы, модели вычислений, алгоритмы: Издательство Нижегородского госуниверситета, 2005.
4. Гергель В.П., Субботина Е.В., Сысоев А.В. Способы организации хранения поисковой информации в программной системе параллельного решения задач глобально-оптимального выбора // Материалы конференции «Технологии Microsoft в теории и практике программирования». 2006. С. 284-288.
5. Казаков А.О. Структуры хранения данных в программной системе параллельного поиска глобально-оптимальных решений Global Expert // Труды конференции «Высокопроизводительные вычисления на кластерных системах». 2008. С. 34-38.