E. S. Kosov, Post-graduate, Tchaikovsky Technology Institute (branch) of Kalashnikov Izhevsk State Technical University

Genetic Algorithm for Researching Focus Advantages of Axial Symmetric Magnetic Fields

The paper considers the ways of job queue coding as applied to a genetic algorithm. The method is proposed to estimate and rank jobs and computation nodes for optimal jobs scheduling and distributing within the set of computational resources.

Key words: genetic algorithm, high-precision focusing, distributed computing, scheduling.

УДК 519.71

Д. Р. Шишов, Ижевский государственный технический университет имени М. Т. Калашникова
А. М. Сметанин, доктор технических наук, профессор, Ижевский государственный технический университет имени М. Т. Калашникова

ПОИСК ОБЪЕКТОВ ИНФОРМАЦИОННОЙ СИСТЕМЫ С ПОМОЩЬЮ РЕШЕНИЯ ЗАДАЧИ О ДОМИНИРОВАНИИ И ФОНОВОГО АЛГОРИТМА

Рассматривается решение задачи о доминировании для оптимизации поиска объектов в дескрипторной информационной системе с использованием графовых моделей. Предлагается алгоритм информационного поиска в фоновом режиме. Предложенные алгоритмы и решения позволяют уменьшить временные затраты на соответствующие операции при задаче восстановления информации.

Ключевые слова: задача о доминировании, фоновый поиск, дескрипторные информационные системы, информационный граф.

Задачу о доминировании (как и задачу интервального поиска) можно отнести к интенсивно развивающемуся сейчас направлению, называемому вычислительной геометрией, связанному с геометрической интерпретацией необязательно геометрических объектов [1]. Многомерная задача о доминировании состоит в поиске в конечном подмножестве n-мерного пространства всех тех точек, которые по каждой из компонент не больше, чем запрос, являющийся точкой данного пространства ($n \ge 1$). Опишем тип задач поиска, соответствующий n-мерной задаче о доминировании.

Пусть $Y_{dom} = [0,1]^n$ и $X_{dom} = [0,1]^n$ – множества записей и запросов соответственно. Пусть на X_{dom} задано вероятностное пространство $\langle X_{dom}, \sigma, P \rangle$, где P задается плотностью распределения вероятностей $p(x)$. Отношение поиска ρ_{dom}, определенное на $X_{dom} \times Y_{dom}$, задается следующим образом:

$$(x_1, \ldots, x_n)\rho_{dom}(y_1, \ldots, y_n) \Leftrightarrow \forall i = \overline{1,n} \;\; y_i \le x_i.$$

Тип $S_{dom} = \langle X_{dom}, Y_{dom}, \rho_{dom}, \sigma, P \rangle$ назовем типом задач о доминировании.

Последовательные алгоритмы решения задачи о доминировании

Пусть

$$G_1 = \left\{ g_{i,\cdot,m}(x_1,\ldots,x_n) = \max\left(1, \rfloor x_i \cdot m \lceil\right) : i \in \{\overline{1,n-1}\} \right\};$$

$$G_2 = \left\{ g_{i,<,a}(x_1,\ldots,x_n) = \begin{cases} 1, & \text{если } x_i < a \\ 2, & \text{если } x_i \ge a \end{cases} : i \in \{\overline{1,n-1}\} \right\};$$

$$F_1 = \left\{ f_{n,\ge,a}(x_1,\ldots,x_n) = \begin{cases} 0, & \text{если } x_n < a \\ 1, & \text{если } x_n \ge a \end{cases} : a \in [0,1] \right\};$$

$$a \in [0,1],\ \tilde{F} = \langle F_1, G_1 \cup G_2 \rangle.$$

Справедливо следующее утверждение.

Если ЗИП $I = \langle X_{dom}, V, \rho_{dom} \rangle$ – n-мерная задача о доминировании, т. е. задача типа S_{dom}, а $|V| = k$, F – базовое множество, определенное выше, $n \ge 1$. Тогда если функция плотности распределения вероятностей ограничена сверху некоторой константой c, то существует граф U объема $Q(U) = C_{k+n-1}^{n} + (3+c) \cdot \sum_{i=1}^{n-1} C_{k+i-1}^{i}$, для которого справедливы следующие оценки сложности:

$$\sum_{y \in V} P\left(O\left(y, \rho_{dom}\right)\right) < T(I, \tilde{F}) \le \sum_{y \in V} P\left(O\left(y, \rho_{dom}\right)\right) + 2n - 1.$$

Данная теорема доказывается в [2].

Приведем теперь примеры алгоритмов, на которых достигаются данные оценки для одномерного и многомерного случаев.

Одномерный случай. Пусть $n = 1$, $V = \{y^1, \ldots, y^k\}$, где $y^i \in [0,1]$, $i = \overline{1,k}$. Будем считать, что записи в V упорядочены в порядке возрастания.

Тогда ИГ U, изображенный на рис. 1, разрешает ЗИП I, и его сложность при объеме $Q(U) = n$ равна

$$T(U) = 1 + \sum_{i=1}^{k-1} P\left(O\left(y^i, \rho_{dom}\right)\right) \le \sum_{i=1}^{k} P\left(O\left(y^i, \rho_{dom}\right)\right) + 1.$$

Получено 27.02.13

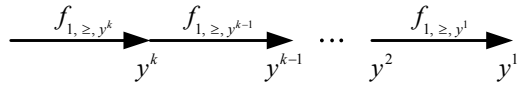

Рис. 1. ИГ для одномерной задачи о доминировании

Многомерный случай. Введем вспомогательные обозначения.

Если $V' \subseteq Y_{dom}$, $r \in [0,1]$ и $1 \le i_1 < ... < i_l \le n$, то обозначим

$$S_r^i(V') = \{y = (y_1,...,y_n) \in V' : y_i \le r\};$$

$$\Pi^{i_1,...,i_l}(V') = \{(y_{i_1},...,y_{i_l}) : (y_1,...,y_n) \in V'\}.$$

Пусть $\tilde{y}^i = (y^1, \ldots, y^i)$, где $i = \overline{0, n-1}$. Здесь и далее под $\tilde{y}^0$ будем понимать пустое место, то есть, например, запись $V^1(\tilde{y}_0)$ – не что иное как V^1.

Пусть M – некоторое конечное, упорядоченное по возрастанию множество точек из отрезка [0, 1]. Пусть $y \in M$. Пусть y'' – следующая за y точка в M, если y – не последняя точка в M, и $y'' = 1$, если последняя. Обозначим

$$Z_y^2(M) = \begin{cases} [y, y''], \text{ если } y - \text{последняя точка в } M, \\ [y, y'') \text{ в противном случае.} \end{cases}$$

Введем для индукции следующие обозначения:

$$V^1 = V, \quad M^1 = \Pi^1(V^1), \quad Y^1 = M^1, \quad X^1 = X_{dom}.$$

Пусть $i \in \{\overline{2, n}\}$, и для всех $j \in \{\overline{1, i-1}\}$ определены множества

$$V^j(\tilde{y}^{j-1}), \quad M^j(\tilde{y}^{j-1}), \quad Y^j, \quad X^j(\tilde{y}^{j-1}),$$

где $\tilde{y}^{j-1} = (y^1, \ldots, y^{j-1})$ – произвольный вектор из Y^{j-1}. Пусть $\tilde{y}^{i-1} = (y^1, \ldots, y^{i-1})$ – произвольный вектор из Y^{i-1}. Тогда обозначим:

$$V^i\left(\tilde{y}^{i-1}\right) = S_{y^{i-1}}^{i-1}\left(V^{i-1}\left(y^1, \ldots, y^{i-2}\right)\right);$$

$$M^i\left(\tilde{y}^{i-1}\right) = \Pi^i\left(V^i\left(\tilde{y}^{i-1}\right)\right);$$

$$Y^i = \left\{\left(y^1, \ldots, y^i\right) : \left(y^1, \ldots, y^{i-1}\right) \in Y^{i-1},\right.$$
$$\left. y^i \in M^i\left(y^1, \ldots, y^{i-1}\right)\right\};$$

$$X^i(\tilde{y}^{i-1}) = \left\{\left(x^1, \ldots, x^n\right) \in X^{i-1}\left(y^1, \ldots, y^{i-2}\right) :\right.$$
$$\left. x_{i-1} \in Z_{y^{i-1}}^2\left(M^{i-1}\left(y^1, \ldots, y^{i-2}\right)\right)\right\}.$$

Построим по индукции некоторый информационный граф U_q ($q \in \{\overline{1, n-1}\}$), который удовлетворяет следующим условиям.

1. Граф U_q имеет $\left|Y^q\right|$ листьев.

2. Между Y^q и множеством листьев графа U_q можно установить взаимно однозначное соответствие такое, что если листу α графа U_q соответствует вектор $(y^1, \ldots, y^q)$ из Y^q, то $N_{\phi_\alpha} = X^{q+1}(y^1, \ldots, y^q)$.

3. $T(U_q) \le 2q$.

4. $Q(U_q) \le (2+c) \cdot \sum_{i=1}^{q} \left|Y^i\right| + \sum_{i=1}^{q-1} \left|Y^i\right| + 1$.

Базис индукции. q = 1. Возьмем множество M^1 и упорядочим его по возрастанию. Построим для него граф U_1.

Положим $m = \left]c \cdot \left|M^1\right|\right[$. Возьмем вершину β_0 и объявим ее корнем графа. Из β_0 выпустим m ребер, припишем им числа от 1 до m, объявим β_0 точкой переключения и припишем ей переключатель $g_{1,\cdot,m}$. Конец ребра с номером i обозначим β_i.

Пусть $V_i = X_i \cap V$, $l_i = \left|V_i\right|$, $i = \overline{1, m}$. Для всех i таких, что $V_i \neq \varnothing$, выполним следующую процедуру. Выпустим из вершины β_i бинарное сбалансированное дерево D_i высоты $\left]\log_2(l_i+1)\right[$ и с $l_i + 1$ концевыми вершинами. Объявим все концевые вершины, кроме первой, листьями и сопоставим слева направо в порядке возрастания элементы из V_i. Для произвольной внутренней вершины β дерева D_i обозначим $y_\beta = \min\limits_{y \in V_{\beta''}} y$, где β'' – конец правого ребра, исходящего из β. Далее объявим все внутренние вершины дерева D_i вершинами переключения и для каждой внутренней вершины β левому ребру, из нее выходящему, припишем 1, правому – 2, а вершине припишем переключатель $g_{1,>,y_\beta}$.

Пусть $i \in \{\overline{1, m}\}$. Обозначим через $j(i)$ индекс ближайшего снизу непустого множества $V_{j(i)}$, т. е. такой номер, что $j(i) < i$, $\left|V_{j(i)}\right| > 0$, не существует j': $\left|V_{j'}\right| > 0$, $j' < i$, $j' > j(i)$. Если такого множества $V_{j(i)}$ нет, то $j(i) = 0$.

Теперь для каждого дерева D_i самую левую вершину дерева отождествляем с самым правым листом дерева $D_{j(i)}$, если $j(i) \neq 0$. Для каждого такого i, что $l_i = 0$, вершину β_i отождествим с самым правым листом дерева $D_{j(i)}$, если $j(i) \neq 0$. Полученный граф и будет графом U_1.

В [2] показано, что граф U_1, разрешающий так называемую вторую задачу о близости, удовлетворяет условиям утверждения индукции.

Индуктивный переход. Пусть $1 < q < n$ и построен граф U_{q-1}, удовлетворяющий условиям индукции. Покажем, как можно построить граф U_q.

Возьмем произвольный лист α графа U_{q-1}. Пусть ему соответствует вектор $(y^1, \ldots, y^{q-1})$ из Y^{q-1}. Упорядочим по возрастанию множество $M^q(y^1, \ldots, y^{q-1})$. Построим для него граф, решающий вторую задачу о близости, как было описано в базисе индукции, где в качестве параметра m возьмем $m = \left\rfloor c \cdot \left| M^q(y^1, \ldots, y^{q-1}) \right| \right\lfloor$.

Заменим переключатели $g_{1,\cdot,m}$ и $g_{1,>,y_\beta}$ на $g_{q,\cdot,m}$ и $g_{q,>,y_\beta}$ соответственно. Полученный ИГ обозначим U_α.

Уберем нагрузку листа α и объявим его обычной вершиной, после чего отождествим его с корнем графа U_α, т. е. U_α будет расти из α. $T(I,F) \geq 2 \cdot \sum_{y \in V} P\left(O\left(y, \succeq^b \right) \right) - t_0$.

Проделаем такую операцию для каждого листа графа U_{q-1} и полученный в результате граф обозначим U_q. Граф U_q также удовлетворяет условиям утверждения индукции [2].

Строить граф U, обеспечивающий мгновенное решение многомерной задачи о доминировании I, будем следующим образом.

Возьмем граф U_{q-1} и рассмотрим произвольный лист α этого графа. Пусть ему соответствует вектор $(y^1, \ldots, y^{q-1})$ из Y^{q-1}. Возьмем множество $V^q(y^1, \ldots, y^{q-1})$ и переобозначим его в V^α. Пусть $\tilde{y}_i = (y_i^1, \ldots, y_i^q)$ $(i = \overline{1,t})$, $V^\alpha = \{\tilde{y}_1, \ldots, \tilde{y}_t\}$, где записи упорядочены по возрастанию последней координаты. Построим граф U'_α, изображенный на рис. 2. Уберем нагрузку листа α и объявим его обычной вершиной, после чего отождествим его с корнем графа U'_α, т. е. U'_α будет расти из α.

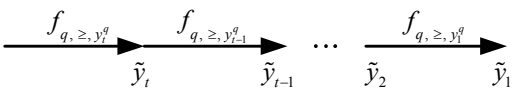

Рис. 2. ИГ U'_α

Проделаем такую операцию для каждого листа графа U_{q-1} и полученный в результате граф обозначим U.

Фоновый алгоритм поиска

При решении задач информационного поиска часто возникают ситуации, когда эти задачи можно решать в так называемом фоновом режиме [3].

Пусть информационная система, осуществляющая резервирование данных предприятия, находится на отдаленной ЭВМ, и доступ к ней осуществляется через связь. Пусть при этом имеется две ЭВМ, одна из которых является основной и осуществляет поиск в базе данных по запросу и подготовку документов из ответа, а вторая является периферийной и отвечает за связь, в частности за отсылку найденных документов абонентам. При этом основная машина может по мере нахождения и подготовки очередного элемента ответа закладывать его в буфер обмена, а периферийная машина может по мере появления в буфере обмена документов отсылать их абонентам. Данный алгоритм можно использовать для оптимизации использования времени, отпускаемого на восстановление информации в случае сбойной ситуации. В качестве временной сложности алгоритма поиска имеет смысл брать суммарное время ожидания очередного элемента ответа пользователем периферийной ЭВМ – во втором. Отсюда видно, что к алгоритмам для работы в фоновом режиме предъявляются другие требования по сравнению с обычными алгоритмами. Для фоновых алгоритмов особенно важна способность быстро получать первый элемент ответа, тогда как все остальные можно получать не так быстро – за отрезки времени, приблизительно равные времени обработки одного элемента ответа пользователем алгоритма.

В качестве модели фоновых алгоритмов поиска предлагается использовать информационные графы (ИГ), но сложность ИГ предлагается вводить по-другому, так, чтобы она отражала сложность фоновых алгоритмов.

Итак, пусть нам даны множество записей Y, множество запросов X, некоторое базовое множество F функций над множеством запросов X и некоторый ИГ U над базовым множеством F. Поскольку для фоновых алгоритмов важны порядок и моменты появления записей в ответе, определим некоторую процедуру обхода графа U, который в дальнейшем поможет определить новое понятие сложности нашего графа.

Чтобы описать эту процедуру, сделаем следующие предположения. Будем считать, что каждому ребру мы можем временно приписывать номер, который назовем путевым; можем отмечать вершины некоторым образом; назовем множество ребер, исходящих из одной вершины, пучком и будем считать, что в каждом пучке определен порядок следования ребер, т. е. существует ребро первое, второе и т. д.

Теперь мы можем перейти к описанию процедуры обхода графа U, которая представляет собой линеаризацию во времени процесса поиска при помощи графа U. Входными данными процедуры обхода B будут информационный граф U и запрос $x \in X$. Выходными данными – упорядоченное множество J записей, входящих в ответ на запрос x, и последовательность T моментов появления записей в ответе.

Итак, пусть нам даны ИГ U и запрос x, тогда процедуру обхода B можно описать следующим образом.

I. (*Инициализация процесса обхода*)

1. Сотрем со всех вершин отметки.

2. Объявим текущей вершиной корень U.

3. Обнулим счетчик времени $t = 0$.

4. Присвоим счетчику путевых номеров значение $m = 1$.

5. Присвоим начальное значение множествам J и T: $J = \varnothing$ и $T = \varnothing$.

6. Перейдем к пункту II.

II. (*Прямой ход*)

1. Если текущая вершина – непомеченный лист графа U, то:

а) заносим в конец множества J запись, принадлежащую текущей вершине;

б) заносим в конец множества T значение счетчика времени *t;*

в) переходим к пункту II.2.

2. Отмечаем текущую вершину.

3. Если текущая вершина есть точка переключения, то:

а) вычисляем значение переключателя, приписанного текущей вершине, на запросе x (пусть это значение равно n);

б) увеличиваем на 1 значение счетчика времени: $t = t + 1$;

в) если среди ребер, исходящих из текущей вершины, нет ребра с номером n, то переходим к пункту III;

г) если конец ребра с номером n – отмеченная вершина, то переходим к пункту III;

е) ребру с номером n, исходящему из текущей вершины, приписываем путевой номер, объявляем конец этого ребра текущей вершиной, присваиваем $m = m + 1$ и возвращаемся к пункту II.

4. Если текущая вершина не является точкой переключения, то:

а) если из текущей вершины не исходит ни одно ребро или из нее исходит ребро, имеющее путевой номер, и это ребро последнее по порядку следования в пучке текущей вершины, то идем к пункту III;

б) если из текущей вершины исходит ребро, имеющее путевой номер, то стираем его с этого ребра и выбираем следующее по порядку следования в пучке ребро; если же из текущей вершины не исходит ребро, имеющее путевой номер, то выбираем первое по порядку следования в пучке текущей вершины ребро;

в) присваиваем выбранному ребру путевой номер m;

г) вычисляем приписанный выбранному ребру предикат на запросе x (пусть его значение равно n);

д) увеличиваем на 1 значение счетчика времени: $t = t + 1$;

е) если $n = 0$, то возвращаемся к пункту II;

ж) если конец выбранного ребра – отмеченная вершина, то возвращаемся к пункту II;

з) если $n = 1$, то объявляем конец выбранного ребра текущей вершиной, присваиваем $m = m + 1$ и возвращаемся к пункту II.

III. (*Обратный ход*)

1. Если из текущей вершины исходит ребро, имеющее путевой номер, то стираем с ребра этот номер.

2. Уменьшаем значение счетчика путевых номеров: $m = m - 1$.

3. Если $m = 0$, то заносим в конец множества T значение счетчика времени и завершаем работу процедуры.

4. Если $m > 0$, то, значит, существует ребро, входящее в текущую вершину и имеющее путевой номер, выбираем его.

5. Объявляем текущей вершиной начало выбранного ребра.

6. Если текущая вершина является точкой переключения, то возвращаемся к пункту III.

7. Иначе возвращаемся к пункту II.

Легко видеть, что данная процедура обхода позволяет посетить все вершины, функции фильтров которых принимают значение 1 на запросе x. Значит, на выходе алгоритма мы получаем множество $J = \{y_{i_1}, ..., y_{i_m}\}$, совпадающее с ответом $I_U(X)$ графа U на запрос x, и множество

$$T = \{t_1, t_2, ..., t_m, t_{m+1}\},$$

где t_i – время появления в ответе i-й записи ($i = \overline{1, m}$); t_{m+1} – время работы процедуры, выраженное в количестве вычисленных предикатов и переключателей. При желании определить его более точно надо сопоставить каждому предикату и переключателю число, равное времени его вычисления. Тогда время работы процедуры будет равно сумме времени вычисления всех переключателей и предикатов. Но мы рассматриваем лишь приближение к реальному времени вычисления, т. е. мы пренебрегаем всеми затратами времени, кроме вычисления предикатов и переключателей, причем считаем, что каждый из них вычисляется за время, равное одному такту.

Применение описанных механизмов поиска информационных объектов позволяет уменьшить временные затраты на соответствующие операции при задаче восстановления информации. В частности, предложен фоновый алгоритм поиска, особенность которого заключается в возможности быстро получать первый элемент ответа, тогда как все остальные можно получать не так быстро – за отрезки времени, приблизительно равные времени передачи по каналу связи одного элемента ответа пользователем алгоритма.

Библиографические ссылки

1. *Хеллман О.* Введение в теорию оптимального поиска. – М. : Наука, 1985.

2. *Гасанов Э. Э.*, *Кудрявцев В. Б.* Теория хранения и поиска информации. – М. : Физматлит, 2002. – 288 с.

3. *Фарли М.* Сети хранения данных. – М. : Лори, 2003. – 550 с.

D. R. Shishov, Post-graduate, Kalashnikov Izhevsk State Technical University
A. M. Smetanin, DSc in Engineering, Professor, Kalashnikov Izhevsk State Technical University

Searching the Information System Objects by Solving the Problem of Domination and Background Algorithm

The paper considers the domination problem solution to optimize the object search within the descriptor information system applying graph models. The algorithm of information search in the background mode is proposed. The proposed algorithms and solutions allow reducing the time required for the corresponding operations within the task of data recovery.

Key words: domination problem, background search, descriptor information systems, information graph.