РЕОПТИМИЗАЦИЯ ПЛАНОВ УПРАВЛЕНИЯ РЕСУРСАМИ В ПОТОКАХ РАБОТ

Ревотюк М. П., Грабовский Д. В.
Кафедра информационных технологий автоматизированных систем, Белорусский государственный университет информатики и радиоэлекторoники
Минск, Республика Беларусь
E-mail: {rmp, kafitas}@bsuir.by

Предлагается модель и инкрементальный алгоритм оптимального управления ресурсами в потоках работ, когда порядок порождаемых деревьев вариантов существенно меньше порядка графа сети работ. Учет текущего состояния сети снижает сложность пересмотра планов до линейной зависимости от объема сканируемого пространства.

I. Постановка задачи

Задачи оптимизации управления потоками работ обычно формулируются в терминах задач о динамическом назначении задач агентам. Агент рассматривается как ресурс, назначаемый для обслуживания заявок. Практически всегда такие задачи сводятся к известным задачам дискретной оптимизации, таким как линейная задача о назначении (ЛЗН) или задача нескольких странствующих коммивояжеров. Однако необходимость учета реальных отношений между агентами и задачами приводит к экспоненциальной сложности алгоритма формирования оптимального назначения. Подобная сложность приводит к задержке момента назначения заданий, снижая эффективность системы агентов. Традиционные приемы использования различного рода аппроксимаций часто неработоспособны из-за недостаточной конкретизации и определенности формируемых решений.

Известно, что классические ЛЗН в виде

$$\begin{cases} Z=\sum\limits_{i=1}^{M}\sum\limits_{j=1}^{N} c_{ij}x_{ij}\to\min \\ \sum\limits_{i=1}^{M} x_{ij}=1, i=\overline{1,M}; \sum\limits_{j=1}^{N} x_{ij}=1, j=\overline{1,N} \end{cases} \tag{1}$$

характеризуются вычислительной сложностью $O(K^3)$, где $K = max(M,N)$. Достаточно часто возникает потребность пересчета задачи (1) после изменения ее исходных данных. Например, при учете моментов времени появления работ или исполнителей, начала и окончания работ можно ставить ЛЗН при изменении состояния портфеля заявок[1]. Варианты ЛЗН отличаются лишь изменением некоторых элементов строки матрицы. Итерация расчета для включаемой строки имеет вычислительную сложность $O(K^2)$, что побуждает использовать наследование результатов предшествующего решения путем его реоптимизации[2,3].

Предмет рассмотрения – способы учета наследования решений ЛЗН для ускорения решения взаимосвязвнных задач [1] в реальном времени. Без потери общности изложение будем вести для случая матричной постановки ЛЗН, однако предлагаемый подход применим и для случая ее графовой постановки. Цель работы - расширение схемы инкрементального алгоритма решения ЛЗН [3] на случай решения потока взаимосвязанных задач.

II. Модель состояния задачи

Преимущества идеи реоптимизации ЛЗН требуют при ее реализации экономичного способа представления области определения задачи. В случае динамических ЛЗН, определенных в матричной форме, можно выделить следующие операции:

включение новых строк или столбцов;

исключение существующих строк или столбцов;

изменение значений элементов матрицы весовых коэффициентов.

Пусть для хранения матрицы весовых коэффициентов выделена память, соответствующая матрице $C(M,N) = (C_{ij}, i = \overline{1,M}, j = \overline{1,N})$. Матрица текущей ЛЗН $c(m,n) = (c_{ij}, i = \overline{1,m}, j = \overline{1,n})$ является подматрицей матрицы $C(M,N)$, но для работы в реальном времени желательно исключение копирования или бесполезной инициализации.

Операции включения и исключения строк или столбцов очевидным образом реализуются на множествах номеров дуг в списке дуг.

Рассмотрим способ выделения подлежащих реоптимизации строк и столбцов.

Пусть в реальном времени формируется поток задач

$$\begin{cases} Z_k=\sum\limits_{i=1}^{m}\sum\limits_{j=1}^{n} c_{ij}^k x_{ij}^k\to\min \\ \sum\limits_{i=1}^{m} x_{ij}^k=1, i=\overline{1,m}; \sum\limits_{j=1}^{n} x_{ij}^k=1, j=\overline{1,n}, \end{cases} \tag{2}$$

нумеруемых индексом $k = 1,2,...$. Прямолинейный подход выявления изменившихся строк и/или столбцов на основе поэлементного сканирования матриц с индексами k и $k+1$ характеризуется вычислительной сложностью $O(MN)$, хотя объем фактических изменений будет $O(mn)$. Для отображения изменения строк матриц будем использовать вектор

$$X^k(i) = k \cdot (c_{ij}^k \equiv c_{ij}^{k-1}), i \in \overline{1,m}, j \in \overline{1,n}, k > 0,$$

а изменения столбцов отобразим вектором

$$Y^k(i) = k \cdot (c_{ij}^k \equiv c_{ij}^{k-1}), i \in \overline{1,m}, j \in \overline{1,n}, k > 0.$$

Если начальное состояние этих векторов $X^0(i) = 0, i = \overline{1,m}$, $Y^0(j) = 0, j = \overline{1,n}$, а матрица коэффициентов ЛЗН при решении задачи минимизации $c_{ij}^0 = \infty, i = \overline{1,m}, j = \overline{1,n}$, то сложность выделения множеств изменения строк и столбцов матриц пропорциональна количеству измененных элементов матрицы. Алгоритм выделения изменений должен формировать стеки индексов изменившихся строк и столбцов.

Алгоритм учета факта изменения $c_{ij}^{k+1} \leftarrow c_{ij}$ в стеке индексов строк $H_x^{k+1}(t)$ и в стеке индексов столбцов $H_y^{k+1}(t)$ на шаге t формирования матрицы с индексами k и $k+1$ имеет вид:

$h_x^{k+1}(0) = 0, h_y^{k+1}(0) = 0$
for $i \in \overline{1,m}, j \in \overline{1,n}$ **do**
  if $c_{ij}^{k+1} \neq c_{ij}$ **then**
    if $X^{k+1}(i) \neq k+1$ **then**
      $h_x^{k+1}(t) \leftarrow h_x^{k+1}(t) + 1;$
      $H_x^{k+1}(h_x^{k+1}(t)) \leftarrow i;$
    end
    if $Y^{k+1}(j) \neq k+1$ **then**
      $h_y^{k+1}(t) \leftarrow h_y^{k+1}(t) + 1;$
      $H_y^{k+1}(h_y^{k+1}(t)) \leftarrow j;$
    end
    $c_{ij} = c_{ij}^{k+1};$
  end
end

Приведенный алгоритм выполняет однократную фиксацию изменения строки или столбца матрицы, а сложность операции сохранения изменившегося элемента матрицы – $O(1)$.

III. Алгоритм решения задачи

Стек индексов строк $H_x^k(t)$ и стек индексов столбцов $H_y^k(t)$ определяют отображение матрицы $c(m,n)$ актуальной ЛЗН на память матрицы $C(M,N)$(при этом $m = \mid H_x^k(t) \mid$ и $n = \mid H_y^k(t) \mid$). Так как строки и столбцы матрицы ЛЗН формально можно не различать, далее будем полагать выбор варианта отображения, когда $m \leq n$.

Наиболее эффективные для решения задачи (1) алгоритмы венгерского метода используют особенности двойственной задачи

$$\begin{cases} Z = \sum\limits_{i=1}^{m} u_i + \sum\limits_{j=1}^{n} v_j \to \max \\ c_{ij} - u_i - v_j \geq 0, i = \overline{1,m}, j = \overline{1,n} \end{cases} \qquad (3)$$

Здесь неизвестными являются потенциалы строк $\{u_i, i = \overline{1,m}\}$ и столбцов $\{v_j, j = \overline{1,n}\}$. Значения потенциалов особого интереса не представляют, но определяют решение задачи (2). Отображение решения (3) будем осуществлять на упорядоченный вариант вектора назначений строк столбцам

$$R_y(j) = \{i \mid c_{ij} - u_i - v_j \geq 0, i = \overline{1,m}, j = \overline{1,n}\}.$$

Схемы известных версий алгоритмов венгерского метода[1-3] совпадают, включая быстрый этап инициализации для формирования начального назначения строк и итерационного дополнения решения для оставшихся строк. Вычислительная сложность этапа инициализации – $O(K^2)$. На этом этапе пытаются выполнить назначение строк, используя операцию приведения матрицы задачи. Приведение состоит в вычитании из элементов столбцов минимальных элементов столбцов. Однако этап инициализации можно исключить, совмещая этап начального назначения строк с этапом последовательного поиска решения для всех оставшихся строк. Такой прием является ключевым для построения инрементального алгоритма реоптимизации [3,4].

Симметричная структура данных модели исключает необходимость транспонирования матриц для соблюдения условия $m \leq n$. Обратное отображение решения (3) на упорядоченный вариант вектора назначений столбцов на строки

$$R_x(i) = \{j \mid c_{ij} - u_i - v_j \geq 0, i = \overline{1,m}, j = \overline{1,n}\}$$

формируется явно либо может поэлементно формироваться неявно любым алгоритмом венгерского метода. Отсюда следует, что векторы $R_x(.)$, $R(.)$, $u_.$ и $v_.$ могут размещаться в предварительно выделенных массивах, размерность которых $K = max(M,N)$. Выбор варианта отображения реализуется проверкой условия $m \leq n$. Переключение между вариантами эдементарно реализуется одношаговым изменением указателей на соответствующие массивы.

Заключение

Таким образом, предложенный прием поиска оптимального паросочетания работ и исполнителей позволяет исключить холостые шаги инициализации переменных состояния или повторения поиска. В результате вычислительная сложность реоптимизации решений ЛЗН линейно зависит от количества измененных кортежей отношения работ и исполнителей.

Список литературы

1. Spivey, M.Z. The Dynamic Assignment Problem/M.Z. Spivey, W.B. Powell//Transportation Science. –2004. –No. 4. –P. 399–419.
2. Toroslu, I.H. Incremental assignment problem/ I.H. Toroslu, G. Üçoluk//Information Sciences. –2007. –Vol.177. –P. 1523-–1529.
3. Ревотюк, М. П. Реоптимизация решения задач о назначении /М. П. Ревотюк, М. П. Батура, А. М. Полоневич //Доклады БГУИР. –2011. – № 1(55). – С. 55–62.
4. Ревотюк, М. П. Быстрая оценка интервалов устойчивости решения линейных задач о назначении/ М. П. Ревотюк, М. К. Кароли, П. М. Батура//Доклады БГУИР. –2013. – № 5(75). – С. 30–36.