Grid 2008

РАСПРЕДЕЛЕНИЕ ВРЕМЕНИ В МНОГОПРОЦЕССОРНОЙ СИСТЕМЕ ПРИ ЛИНЕЙНЫХ ФУНКЦИЯХ ПОЛЕЗНОСТИ

С.В. Бредихин, А.Б. Хуторецкий, Е.Б. Шумило

По мере увеличения круга пользователей и развития вычислительных систем возникает проблема покрытия эксплутационных расходов и затрат на их развитие. В настоящее время популярна идея вычислительных систем, предоставляющие платные услуги. Проблема финансирования таких систем пока не решена. В идеальном случае можно было бы в механизм управления ресурсами встроить подсистему, вычисляющую цену равновесия. Тогда продажа ресурсов будет производиться по вычисленным ценам равновесия, и это решает проблему платности.

Grid 2008

Содержание доклада

- Структура рынка
- Модель поставщика
- Модель пользователя
- Равновесия
- Расписания
- Результаты
- Идея алгоритма построения равновесного расписания

Grid 2008

Структура рынка

Мы рассматриваем рынок, на котором продаются такты различных процессоров на период длины T.

На рынке M потребителей (пользователей) и N поставщиков, каждый из которых владеет одним процессором.

Процессоры могут иметь различную производительность, владельцы процессоров могут иметь различные отправные цены.

Каждый пользователь имеет линейную функцию полезности, и в течение рассматриваемого периода хотел бы выполнить одну задачу.

Задача может использовать различные процессоры, но не одновременно.

2. Модель поставщика

Поставщик j владеет процессором, который за единицу времени выполняет s_j тактов с удельными затратами c_j.

Всего за период от может поставить $s_j T$ тактов.

Продавая q_j тактов по цене p_j, поставщик получит прибыль $(p_j - c_j)q_j$. Его цель – выбрать объем q_j предложения за период так, чтобы максимизировать прибыль: $(p_j - c_j)q_j \to \max$

при условии $0 \le q_j \le s_j T$.

2. Модель поставщика (продолжение)

Тогда отображение предложения поставщика *j* имеет вид:

$$S_j(p_j) = \begin{cases} 0, \ if \ \ p_j < c_j, \\ [0, s_j T\,], \ if \ \ p_j = c_j\ , \\ s_j T, \ if \quad p_j > c_j . \end{cases}$$

3. Модель пользователя

Пользователь i имеет бюджет B_i и задачу объема Q_i. За один такт он, следовательно, готов заплатить $a_i = B_i / Q_i$.

Потребительский набор пользователя i – это вектор $\mathbf{x_i} = (x_{ij})_j$, где $x_{ij} \geq 0$ – число тактов процессора j. Вектор $\mathbf{x_i}$ должен удовлетворять следующим условиям:

$$\sum_j \frac{x_{ij}}{s_j} \leq T$$

(суммарное время выполнения задачи i на всех процессорах не должно превышать длину периода);

$$\sum_j x_{ij} \leq Q_i$$

3. Модель пользователя (продолжение)

(суммарное число тактов, выделенное задаче на всех процессорах, не должно превышать ее объем). Пусть X_i – множество всех потребительских наборов пользователя i. Для векторов $\mathbf{x} \in X_i$ определена функция полезности пользователя

$$v_i(\mathbf{x_i}) = \sum_j (a_i - p_j)x_{ij}.$$

Пусть p_j цена такта процессора j, $\mathbf{p} = (p_j)_j$. При ценах $\mathbf{p}$ пользователь выбирает потребительский набор $\mathbf{x_i} \in X_i$ так, чтобы максимизировать функцию полезности $v_i(\mathbf{x_i})$ при бюджетном ограничении

$$\sum_j p_j x_{ij} \le B_i$$

4. Равновесия

Если предложение поставщика j равно q_j, то можно считать, что он предъявляет спрос на остальные $x_{oj} = s_j T - q_j$ тактов. Пусть $\mathbf{x_o} = (x_{oj})_j$.

Равновесием на рассматриваемом рынке называется набор $(\mathbf{x}, \mathbf{p})$, где $\mathbf{x} = (\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n)$, такой, что $\mathbf{x}_i$ максимизирует функцию полезности $v_i(\mathbf{x}_i)$ на множестве X_i, $\mathbf{p} \geq \mathbf{0}$ и выполнено условие

$$\sum_{i=0}^{n} x_{ij} \leq s_j T,$$

причём при $p_j > 0$ достигается равенство (избыточное предложение имеет нулевую цену).

Если $(\mathbf{x}, \mathbf{p})$ равновесие, то будем говорить, что $\mathbf{p}$ – *вектор цен равновесия*, а $\mathbf{x}$ – *равновесное распределение*.

5. Расписания

Расписание, соответствующее распределению **x** – это совокупность таких промежутков времени $[\alpha_{ij}^{k}, \beta_{ij}^{k}]$ для каждого i, $1 \le i \le N$ (в течение которых запланировано выполнение задачи i на процессоре j), что:

(1) $[\alpha_{ij}^{k}, \beta_{ij}^{k}] \cap [\alpha_{mn}^{l}, \beta_{mn}^{l}] = \varnothing$,

если $j = n$ и $i \neq m$, или $j = n$, $i = m$ и $k \neq l$, или $j \neq n$ и $i = m$

(промежутки, выделенные разным задачам на одном процессоре, попарно не пересекаются и промежутки, выделенные одной задаче, тоже попарно не пересекаются);

(2) $S_{ij} \sum_{k} [\beta_{ij}^{k} - \alpha_{ij}^{k}] = x_{ij}$

(задаче i на процессоре j выделено суммарное время, обеспечивающее выполнение x_{ij} тактов).

Не всякому распределению соответствует расписание. Расписание, соответствующее равновесному распределению (если оно существует), будем называть *равновесным*.

6. Результаты

1. Все равновесные распределения могут быть получены как решения некоторой задачи линейного программирования, а цены равновесия могут быть определены на основе двойственных оценок ограничений той же задачи.

2. В предположении, что наиболее трудоемкая работа может быть выполнена за распределяемое время на самом медленном процессоре, предложен простой алгоритм для построения расписания, которое соответствует некоторому равновесному распределению.

7. Идея алгоритма построения равновесного расписания

Упорядочим процессоры по неубыванию величин c_j, а задачи – по невозрастанию величин a_i.

Если $a_1 < c_1$, то «пустое» расписание, в котором ни одна задача не решается, является равновесным.

Если $a_1 \geq c_1$, то на каждом шаге делаем следующее: выбираем первый недогруженный процессор j и первую не полностью размещенную задачу i. Если $a_i \geq c_j$, то выделяем задаче i на процессоре j число тактов, равное минимуму из неразмещенного объема задачи и нераспределенной мощности процессора. Если $a_i < c_j$, или $i = M$, или $j = N$, то расписание построено.

7. Идея алгоритма построения равновесного расписания (продолжение)

Из предположения (слайд 6, п. 2) следует, что

(а) алгоритм выделит каждой задаче не более двух промежутков (возможно, ни одного);

(б) если задаче выделены два промежутка (α и β), то они не пересекаются, размещены на процессорах с соседними номерами (*j* и *j* + 1), причем промежуток α занимает на процессоре *j* конец распределяемого периода, а промежуток β занимает на процессоре *j* +1 начало периода.

Благодарю за внимание

bred@nsc.ru