Przetwarzanie równoległe
Zadanie domowe I

Jarosław Marek Gliwiński
#indeksu 74839

31 października 2009

1 Wstęp

W ramach rozwiązania rozpatrzyłem kilka przypadków. Pełne rozwiązanie prezentuję jedynie dla przypadku, który uznałem za ostateczne rozwiązanie zadania. Pozostałe przypadki zostały potraktowane w sposób mniej ścisły bądź pominięte. Wynika to zarówno z ograniczeń czasowo-objętościowych, jak i nieistotności tych rozważań dla ostatecznego rozwiązania.

2 Analiza wariantów kolejności działań w systemie

2.1 Wariant pierwszy

Rozpoczynamy od założenia, że wszystkie cykle przetwarzania są jednakowe. Jedyną aktywnością systemu na początku cyklu przetwarzania może być pobieranie danych ze źródła przez procesor P_1. Pozostałe dwa procesory nie mogą przetwarzać danych bez otrzymania ich od P_1, tak samo on sam nie może jednocześnie wysyłać i pobierać tych samych danych. Przy takim założeniu należy doliczyć do czasu w cyklu każdego procesora czas oczekiwania na ten proces $S_1 + C_1V$. Następnym etapem jest przesyłanie danych pobranych ze źródła do P_2 i P_3. W związku z tym odpowiednio należy doliczyć czasy $S_2+C_2V_2$ i $S_3+C_3V_3$ konieczne na odbiór danych do przetworzenia w tych procesorach. Przesył na obydwu tych łączach może odbywać się równolegle, toteż do cyklu procesora P_1 należy doliczyć dłuższy z tych czasów. Tak więc podstawą opisu matematycznego takiego systemu będą następujące ograniczenia:

$$S_1 + C_1V + A_1V_1 + \max[S_2 + C_2V_2, S_3 + C_3V_3] \leqslant 1000 \tag{2.1.1}$$

$$S_1 + C_1V + A_2V_2 + S_2 + C_2V_2 \leqslant 1000 \tag{2.1.2}$$

$$S_1 + C_1V + A_3V_3 + S_3 + C_3V_3 \leqslant 1000 \tag{2.1.3}$$

Rzecz jasna należy przy tym uwzględnić pewne dodatkowe ograniczenia oraz, rzecz jasna, funkcję celu. W tym przypadku funkcją celu będzie liczba przetworzonych jednostek danych $V = V_1 + V_2 + V_3$, przy czym V_1,V_2,V_3 to jednostki przyporzadkowane poszczególnym procesorom, i będzie ona podlegała maksymalizacji: $\max V$. Pozostałe ograniczenia, jak i postać pliku wejściowego `lpsolve` podane będą przy rozwiazaniu końcowym.

2.2 Wariant drugi

Łatwo w poprzednim rozwiązaniu zauważyć, że gdyby możliwe było równoległe wykonanie wszystkich trzech komunikacji, zredukowany zostałby czas bezczynności każdego z procesorów. Rozwiązanie takie wymaga jednak nie tyle złamiania, a nagięcia założenia o identyczności cykli przetwarzania. Jeśli wyobrazimy sobie przypadek, w którym pierwszy cykl przeznaczymy na wstępne załadowanie danych do P_1, zaś kolejne będą przebiegały analogicznie do poprzedniego przypadku, z jedną – kluczową – różnicą. Mianowicie, teraz w obrębie jednego cyklu dane pobierane przez P_1 ze źródła będą o jeden cykl „nowsze" niż przesyłane do P_2 i P_3. Tym samym równoległe wykonanie wszystkich trzech komunikacji staje sie możliwe, jako że w ramach żadnej z nich nie są przesyłane jednocześnie te same jednostki danych. Równania w tym potokowym systemie będą się prezentowały następująco (tym razem w pełnej postaci):

$$\max V \tag{2.2.1}$$

$$A_1V_1 + \max[S_2 + C_2V_2, S_3 + C_3V_3, S_1 + C_1V] \leqslant 1000 \tag{2.2.2}$$

$$A_1V_1 + \max[S_2 + C_2V_2, S_3 + C_3V_3, S_1 + C_1V] \leqslant 1000 \tag{2.2.3}$$

$$A_2V_2 + S_2 + C_2V_2 \leqslant 1000 \tag{2.2.4}$$

$$A_3V_3 + S_3 + C_3V_3 \leqslant 1000 \tag{2.2.5}$$

$$V = V_1 + V_2 + V_3 \tag{2.2.6}$$

$$V_1, V_2, V_3 \in \mathbb{Z}_+ \tag{2.2.7}$$

Gwoli wyjaśnienia wprowadzonych zmian w stosunku do równań 2.1.1 – 2.1.3, należy zaznaczyć, co nastepuje:

1. przesunięcie członu $S_1 + C_1V$ w 2.1.1 wynika z pełnej równoległości wykonywanych z udziałem P_1 komunikacji
2. zniknięcie $S_1 + C_1V$ z dwóch pozostałych równań wynika pośrednio z tego samego, a bezpośrednio ze braku konieczności oczekiwania na komunikację źródło$\rightarrow P_1$

2.3 Dalsze rozważania

Być może możliwa byłaby dalsza optymalizacja systemu przy pełnej rezygnacji z założenia o identyczności cykli przetwarzania. Pozwoliłby to na wprowadzenie powtarzających sie sekwencji kilku cykli (każda sekwencja tworzyłaby coś w rodzaju „nadcyklu" przetwarzania), gdzie wynikiem byłoby uśrednienie wyników po całej sekwencji.

Można wyobrazić sobie także (wykorzystując założenie o niewyczerpanej pamieci procesorów) system, w którym najpierw P_1 pobiera ze źródła dowolnie dużo danych (w ciągu dużej

liczby cykli $1000ms$), a następnie przesyła jedynie do pozostałych procesorów dane, które nadążą one przetworzyć w każdym kolejnym cyklu „drugiej fazy". Dokonalibyśmy w ten sposób podziału na dwie klasy cykli, które następowałyby po sobie w dwóch długich ciągach, w teoretycznym przypadku granicznym dążących do nieskończoności.

Głębsze rozważania na ten temat zaniedbałem, jakkolwiek uważam, że obydwa te podejścia mają potencjał do uzyskania wyników co najmniej porównywalnych z otrzymanym na podstawie analizy z punktu 2.2.

3 Wyniki

Jak zaznaczyłem we wstepie, pełne rozwiązanie zostanie zaprezentowane jedynie dla jednego przypadku (wariantu z punktu 2.2) uznanego za najlepiej odpowiadający jednocześnie warunkom zadania jak i kryteriom optymalizacji.

3.1 Postać ograniczeń w formacie `lpsolve`

Pozorne niekonsekwencje w zapisie wynikają z ograniczeń formatu akceptowanego przez program. W szczególności zmienną m i powiaząnymi z nią ograniczeniami zastąpiono funkcję maksimum. Konieczne było też zastąpienie stałych wartościami liczbowymi.

```
/* Objective function */
max: v;
/* Variable bounds */
m >= 10 10 v2;
m >= 10 5 v3;
m >=  10 2 v1 ;
v = v1 v2 v3;
1000 >= 10 v1 m;
1000 >= 10 10 v2 10 v2;
1000 >= 10 5 v3 20 v3;
int v1 v2 v3;
```

3.2 Liczbowe wyniki optymalizacji

Wyniki podane w jednostkach przetworzonych przez cały system oraz przetworzonych przez każdy z procesorów przedstawia poniższa tabela.

Parametr	Wartość
V	138
V_1	79
V_2	20
V_3	39

3.3 Wykorzystanie procesorów i prędkość przetwarzania systemu

Prędkość przetwarzania systemu przedstawiona została w tabeli w punkcie poprzednim – to 138 jednostek na cykl, tj. na $1000ms$. Wykorzystanie procesora można mierzyć dwojako – najbardziej oczywistą miarę jest uwzględniająca jedynie czas przetwarzania, jednak można

sobie wyobrazić również taką uznającą za czas zajętości (wykorzystania) procesora także czas przeznaczony na komunikację.

Użyłem miary „oczywistej", w której czas poświęcony na komunikację jest uznawany za bezczynność, jako że z punktu widzenia obliczeń nie wprowadza w nich postępu. Precyzując: obliczenia zostały wykonane przy pomocy wzoru $\frac{V_i \dot{A}_i}{T}$. W praktyce oznacza to procent ilości danych, które teoretycznei mogłyby zostać przetworzone, gdyby dany procesor przez cały okres T nieustannie przetwarzał dane.

Procesor	Wykorzystanie [%]
P_1	79
P_2	20
P_3	78

Wykaz skrótów i oznaczeń

Wartości liczbowe zaczerpnięte zostały z treści zadania, toteż nie przytaczam ich.

V liczba jednostek danych przetwarzanych w cyklu

P_i i-ty procesor

V_i liczba jednostek danych przetwarzanych w cyklu na i-tym procesorze

A_i prędkość przetwarzania na i-tym procesorze

C_i odwrotność prędkości komunikacji do i-tego procesora

S_i składowa stała komunikacji do i-tego procesora

T długość cyklu przetwarzania