

Semestrální projekt MI-PDP 2020/2021:

Paralelní algoritmus pro řešení problému

Martin Šafránek

magisterské studium, FIT CVUT, Thakurova 9, 160 00 Praha 6

April 22, 2021

1 Definice problému a popis sekvencního algoritmu

Program řeší problém nalezení optimální posloupnosti tahů pro střelce a jezdce, která vede k sebrání všech pěšců rozmístěných na šachovnici. Jedná se o analogii problému obchodního cestujícího. Nalezení optimálního řešení je proto NP těžký úkol. Řešení v této práci používá bruteforce s heuristikami pro ořezávání stavového prostoru.

Příklad vstupu je na obrázku 1. Obsahuje vždy

1. přirozené číslo k , reprezentující délku strany šachovnice S o velikosti $k \times k$,
2. pole souřadnic rozmístěných figurek na šachovnici S ,
3. horní mez délky optimální posloupnosti d_{max}^* .

```
11
22
-----
-----P---
P-----P--P-
---PJS--P--
----P-----
---P--P----
-----P----
-----P----
-----P----
-----PP----
-----
```

Obrázek 1: Příklad vstupních dat pro $k = 11$, $d_{max}^* = 22$. Střelec je označen S, jezdec J, pěšák P a prázdné políčko -.

Sekvenční algoritmus je popsán v 1. Používá dvě heuristiky.

Heuristika střelec. Z množiny možných políček, kam je možné střelce přemístit jsou preferována ty, která obsahují pěšce. Pokud takové políčko neexistuje, jsou preferována políčka s alespoň jedním pěšákem na diagonále. Jinak se pohyb střelce rozhodne náhodně.

Heuristika kůň. Z množiny možných políček, kam je možné koně přemístit jsou preferována ty, která obsahují pěšce. Pokud takové políčko neexistuje, jsou preferována políčka, z kterých kůň ve svém následujícím tahu může vzít pěšáka. Pokud ani takové políčko neexistuje, jsou preferována políčka, z kterých kůň v následujících dvou tazích může vzít pěšáka. Jinak se pohyb koně rozhodne náhodně.

```

input :  $k \times k$  pole, mez  $d_{max}^*$ 
output: optimální posloupnost tahů
1 if abc or def then
2 else
3 | ;
4 end
5 while While condition do
6 | instructions;
7 | if condition then
8 | | instructions1;
9 | | instructions2;
10 | else
11 | | instructions3;
12 | end
13 end

```

Algoritmus 1: sekvenční

2 Popis paralelního algoritmu a jeho implementace v OpenMP - taskový paralelismus

Popiste paralelní algoritmus, opět vyjdete ze zadání a přesně vymeďte odchylky, které při implementaci OpenMP používáte. Popiste a vysvětlete strukturu celkového paralelního algoritmu na úrovni procesů v OpenMP a strukturu kódu jednotlivých procesů. Např. jak je naimplementována smyčka pro činnost procesu v aktivním stavu i v stavu nečinnosti. Jaké jste zvolili konstanty a parametry pro skalování algoritmu. Struktura a semantika příkazové řádky pro spouštění programu.

3 Popis paralelního algoritmu a jeho implementace v OpenMP - datový paralelismus

Popiste paralelní algoritmus, opět vyjdete ze zadání a přesně vymeďte odchylky, které při implementaci OpenMP používáte. Popiste a vysvětlete strukturu celkového paralelního algoritmu na úrovni procesů v OpenMP a strukturu kódu jednotlivých procesů. Např. jak je naimplementována smyčka pro činnost procesu v aktivním stavu i v stavu nečinnosti. Jaké jste zvolili konstanty a parametry pro skalování algoritmu. Struktura a semantika příkazové řádky pro spouštění programu.

4 Popis paralelního algoritmu a jeho implementace v MPI

Popiste paralelní algoritmus, opět vyjdete ze zadání a přesně vymeďte odchylky, zvláště u Master-Slave části. Popiste a vysvětlete strukturu celkového paralelního algoritmu na úrovni procesů v MPI a strukturu kódu jednotlivých procesů. Např. jak je naimplementována smyčka pro činnost procesu v aktivním stavu i v stavu nečinnosti. Jaké jste zvolili konstanty a parametry pro skalování algoritmu. Struktura a semantika příkazové řádky pro spouštění programu.

5 Namerené výsledky a vyhodnocení

1. Zvolte tři instance problému s takovou velikostí vstupních dat, pro které má sekvenční algoritmus časovou složitost mezi 1 a 10 minutami. Pro měření času potřebný na čtení dat z disku a uložení na disk neuvazujte a zakomentujte řádky tisky, logy, zprávy a výstupy.
2. Měřte paralelní čas při použití $i = 2, \dots, 60$ výpočetních jader.
3. Tabulková a případně graficky zpracované namerené hodnoty časové složitosti měřených instancí běhu programu s popisem instancí dat. Z namerených dat sestavte grafy zrychlení $S(n, p)$.

4. Analyza a hodnoceni vlastnosti paralelniho programu, zvlaste jeho efektivnosti a skalovatelnosti, pripadne popis zjisteneho superlinearniho zrychleni.

6 Zaver

Celkove zhodnoceni semestralni prace a zkusenosti ziskanych behem semestru.

7 Literatura