Zadání projektů z předmětu **Algoritmy I**
letní semestr 2021/2022
prezenční studium

Verze zadání

4. dubna 2022 První verze

Obecné pokyny

1. Celkem je k dispozici 6 zadání projektů.
2. Každý student má přiděleno jedno zadání.
3. **Termín odevzdání je 8. května 2022 ve 23:59**. Tento termín je konečný a nebude dále posunován.
4. Každý cvičící Vám sdělí, jakým způsobem budete projekt odevzdávat – emailem, uložením na sdílené úložiště, pomocí ftp, repozitáře GitHub a tak podobně.
5. Součástí zdrojových kódů Vašeho programu bude programátorská dokumentace ve formě dokumentačních komentářů, zpracovatelných programem Doxygen, viz `www.doxygen.org`. Vygenerovanou dokumentaci není nutné odevzdávat, postačuje pokud Vámi odevzdaný archiv bude obsahovat konfigurační soubor `doxyfile`, případné adresáře pro vygenerovanou dokumentaci, vkládané obrázky atd.
6. Termíny obhajoby budou vypsány v systému Edison.
7. Řešení projektu bude hodnoceno podle následujících kritérií:
   (a) správnost řešení,
   (b) způsob implementace (rozložení implementace do funkcí, hlavičkových a zdrojových souborů a tak dále),
   (c) způsob zápisu, čitelnost zdrojových kódů (odsazování, pojmenování proměnných, funkcí a tak dále) a
   (d) efektivita implementovaného algoritmu[1].

   Správnost řešení je podmínka nezbytná, aplikace, která nebude poskytovat správné výsledky bude hodnocena automaticky 0 body.
8. **Použité zdroje je nutné správně citovat.** K řešení můžete používat odbornou literaturu, učebnice, příklady zdrojových kódů z ostatních předmětů, internetové zdroje. V tom případě je nutné uvést, že „Tento algoritmus jsem převzal z…“, „Tuto část kódu jsem převzal z…“. Tyto případné citace umístíte do dokumentačních komentářů.
9. S případnými dotazy se obracejte primárně na svého cvičícího.

[1] Smyslem tohoto kritéria není nutit Vás k implementaci nejlepšího známého algoritmu. Tímto kritériem si ponecháváme prostor pro případné snížení bodového hodnocení za použití algoritmu zcela nevhodného, nesmyslného, zmateného.

1 Scheduler procesoru

1.1 Problém

Moderní SoC (System-on-a-chip) založené na architektuře ARM mají typicky tři druhy jader, která jsou různě výkonná:

- Cortex-X2 s vysokým výkonem,
- Cortex-A710 se středním výkonem a
- Cortex-A510 s nízkým výkonem.

Každé z těchto jader je vhodné pro různě náročné, tedy typově rozdílné, výpočetní úlohy. Uvažujme, že výpočetní čas úlohy závisí rovněž na typu jádra, na kterém výpočet probíhá. Nejde totiž jen o výkon, ale také o režijní náklady, při volbě nevhodného typu jádra. Rozlišujeme tři typy úloh:

- Náročná (H) – trvá 1 jednotku času na vysoce výkonném jádru (Cortex-X2), 2 jednotky na ostatních,
- Středně náročná (M) – trvá 1 jednotku času na středně výkonném jádru (Cortex-A710), 2 jednotky na ostatních,
- Nenáročná (L) – trvá 1 jednotku času na jádru s nízkým výkonem (Cortex-A510), 2 jednotky na ostatních.

Úlohy se zpracovávají po dávkách (tzv. batch), kde každá dávka obsahuje 8 úloh. V každém kroku je dávka 8 úloh sestavena, následně rozdělena mezi jádra a zpracována. Po zpracování všech úloh z dané dávky se pokračuje analogicky s další dávkou 8 úloh. Každé jádro má frontu úloh, které má zpracovat, zpracování probíhá paralelně. Dávka úloh je považována za zpracovanou po dokončení všech úloh z dané dávky.

Implementujte funkce, resp. metody tříd, které umožní provést následující dvojici simulací.

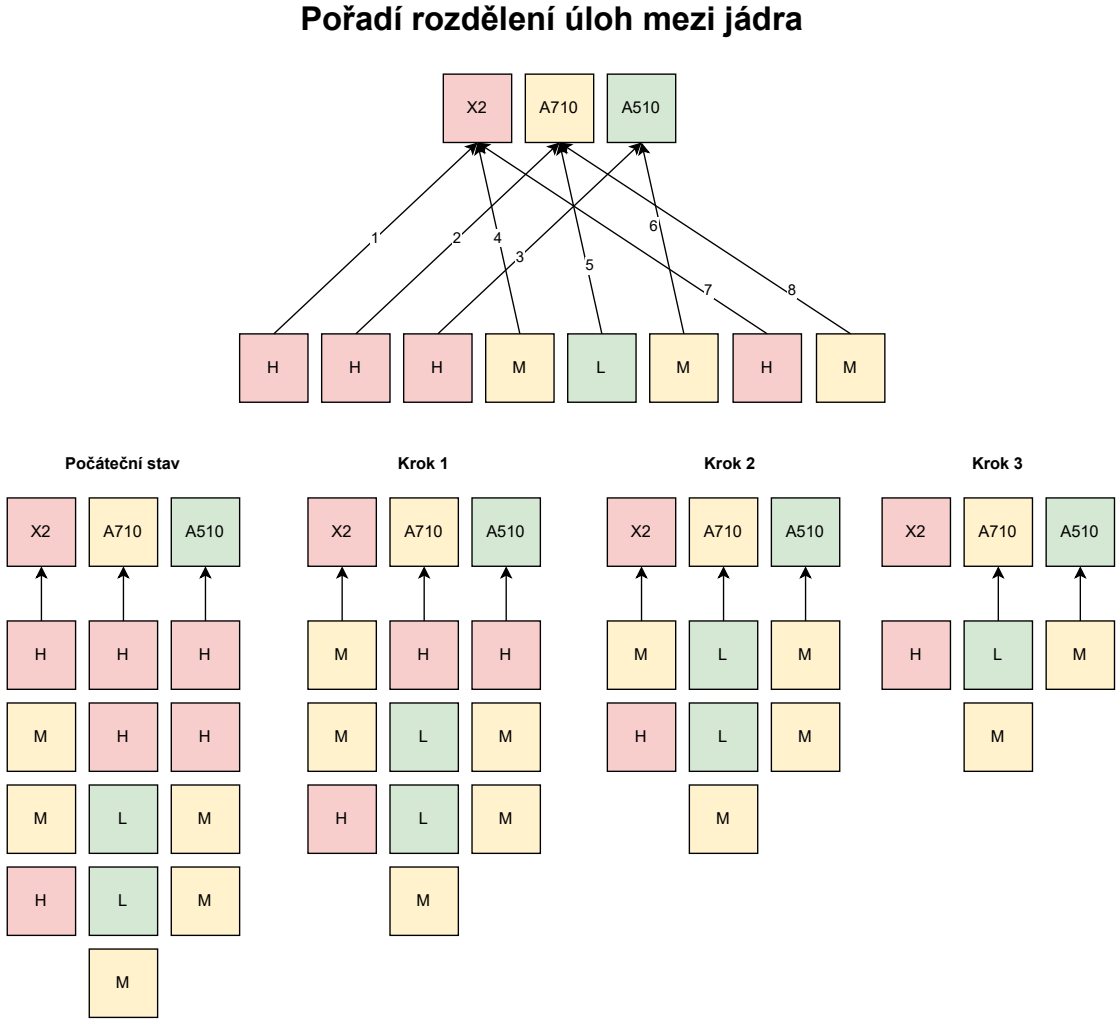
- V první simulaci jsou úlohy z dávek rozděleny systémem round-robin[2]. To znamená, že první úloha je přiřazena vysoce výkonnému jádru, druhá středně výkonnému, třetí jádru s nízkým výkonem, čtvrtá vysoce výkonnému, pátá středně výkonnému a tak dále, až do rozdělení celé dávky. Úlohy jsou tedy rozděleny bez ohledu na jejich typ a vhodnost jádra. Vašim úkolem je provést simulaci tohoto scénáře a zjistit, kolik jednotek času celá simulace zabrala.
- Ve druhé simulaci je vaším úkolem navrhnout systém, dle kterého budou úlohy z dávky rozděleny mezi jádra lepším způsobem, to jest tak, aby došlo ke snížení času nutného pro zpracování dávky úloh[3]. Simulaci proveďte a zjistěte, kolik jednotek času celá simulace zabrala. Porovnejte čas s první simulací.

1.2 Ukázka

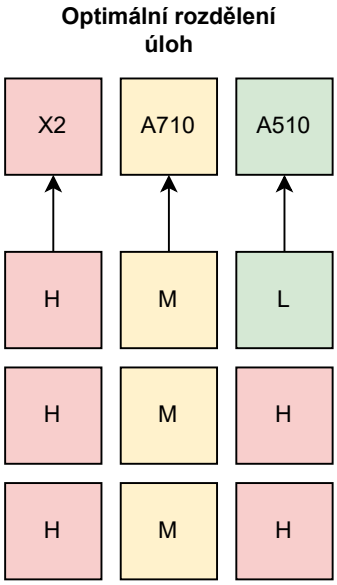
Na obrázku 1 vidíme ukázku rozdělení úloh pomocí systému round-robin a první 3 kroky zpracování. Všimněte si, že v případě přiřazení úlohy k nevhodnému jádru se položka ve frontě zdvojuje, čímž je simulován dvojnásobný čas zpracování. Zpracování této dávky zabere 5 jednotek času. Na obrázku 2 vidíme rovněž příklad lepšího rozdělení úloh, kdy zpracování dávky úloh zabere pouze 3 jednotky času.

[2] V češtině bychom jej mohli nazvat „chodí pešek okolo“.

[3] Samozřejmě, v některých případech může být rozdělení úloh systémem round-robin nejlepší možné. Ale většinou tomu tak nebude.



Obrázek 1: Diagram popisující simulaci rozdělení úloh systémem round-robin



Obrázek 2: Diagram s ukázkou lepšího rozdělení

1.3 Implementace

- Vstupem je textový soubor. Na prvním řádku je uveden celkový počet úloh v souboru.
- Na každém následujícím řádku je pouze jeden znak, který označuje typ úlohy (H, M nebo L).
- Vstupní soubor s jednou dávkou úloh může vypadat například následovně:

```
8
H
H
H
M
L
M
H
M
```

2 Klokan

2.1 Problém

Představme si jednoduchou hru pro jednoho hráče, kterou si můžeme pojmenovat třeba „Klokan“. Základem této hry je posloupnost $P = p_0 p_1 \dots p_{n-1}$, která obsahuje n číslic 0 až 9, číslice se mohou opakovat. Matematicky řečeno $p_i \in \{0, 1, \dots, 9\}$ pro $i = 0, 1, \dots, n-1$.

Po posloupnosti P skáče klokan a předpokládejme, že se klokan nachází na pozici i. Klokan může provést následující skoky:

1. může přeskočit na pozici vlevo, tedy na pozici $i - 1$, pokud tato pozice existuje,