Tematy, zadania i pytania egzaminacyjne "Wydajność oprogramowania"

A

1. Podaj przykład kodu, w którym o czasie wykonania będzie decydować opóźnienie (*latency*) przy wykonywaniu przez potoki przetwarzania konkretnej operacji arytmetycznej. Jakich wartości CPI należy się spodziewać w takim przypadku? Jak przełoży się to na wydajność w Gflop/s?

2. Podaj przykład kodu, w którym o czasie wykonania będzie decydować przepustowość (*throughput*) przy wykonywaniu przez potoki przetwarzania konkretnej skalarnej operacji arytmetycznej. Jakich wartości CPI należy się spodziewać w takim przypadku? Jak przełoży się to na wydajność w Gflop/s?

3. Podaj przykład kodu, w którym o czasie wykonania będzie decydować przepustowość (*throughput*) przy wykonywaniu przez potoki przetwarzania konkretnego rozkazu wektorowego odpowiadającego operacji arytmetycznej. Jakich wartości CPI należy się spodziewać w takim przypadku? Jak przełoży się to na wydajność w Gflop/s?

4. Podaj przykład kodu, w którym występuje zależność danych dla kolejnych iteracji pętli. W jaki sposób wpływa ona na wydajność przetwarzania.

5. Podaj przykład kodu, w którym o czasie wykonania będzie decydować opóźnienie (*latency*) pobierania danych z pamięci. Jakie cechy kodu decydują o tym, że procesor nie jest w stanie wykorzystać sprzętowych technik ukrywania opóźnienia (scharakteryzuj dla konkretnych technik ukrywania opóźnienia)?

6. Podaj przykład kodu, w którym o czasie wykonania będzie decydować przepustowość (*throughput*) pobierania danych z pamięci. Jakie cechy kodu decydują o tym, że procesor jest w stanie wykorzystać sprzętowe techniki ukrywania opóźnienia (scharakteryzuj dla konkretnych technik ukrywania opóźnienia)?

7. Podaj przykład kodu, który może posłużyć do ustalenia rozmiaru pamięci podręcznych poszczególnych poziomów w mikroprocesorze. Omów zasadę pomiaru.

8. Podaj przykład kodu, który może posłużyć do ustalenia drożności pamięci podręcznej L1 w rdzeniu mikroprocesora. Omów zasadę pomiaru.

9. Podaj przykład kodu, który może posłużyć do ustalenia rozmiaru linii pamięci podręcznej L1 w rdzeniu mikroprocesora. Omów zasadę pomiaru.

10. Podaj przykład kodu, w którym może wystąpić zjawisko fałszywego współdzielenia przy wykonaniu wielowątkowym. Opisz jak przebiega wykonanie kodu w takim przypadku (w szczególności jak funkcjonuje pamięć podręczna) i jak można zmodyfikować kod, aby uniknąć fałszywego współdzielenia.

11. Podaj przykład kodu operującego na tablicach dwuwymiarowych (macierzach), w którym zastosowano optymalizację *array padding*. Załóż, że pamięć dla elementów macierzy jest alokowana statycznie w postaci standardowej tablicy dwuwymiarowej języka C. Uwzględnij w kodzie alokację pamięci dla tablicy i pętlę, w której odwiedzane są wszystkie elementy tablicy.

12. Podaj przykład kodu operującego na tablicach dwuwymiarowych (macierzach), w którym zastosowano optymalizację *array padding*. Załóż, że pamięć dla elementów macierzy jest alokowana dynamicznie za pomocą funkcji *malloc*. Uwzględnij w kodzie alokację pamięci dla tablicy i pętlę, w której odwiedzane są wszystkie elementy tablicy.

13. Na czym polegają klasyczne optymalizacje *constant folding, copy propagation, strength reduction, common subexpression elimination*? Podaj przykłady optymalizacji i ich wpływ na wydajność

14. Na czym polegają klasyczne optymalizacje dotyczące pętli: *loop invariant code motion* i *induction variable simplification*? Podaj przykłady optymalizacji i ich wpływ na wydajność

15. Na czym polegają klasyczne optymalizacje dotyczące pętli: *loop fusion* i *loop fission*? Podaj przykłady optymalizacji i ich wpływ na wydajność

16. Na czym polegają klasyczne optymalizacje dotyczące pętli: *loop interchange* i *loop unrolling*? Podaj przykłady optymalizacji i ich wpływ na wydajność

17. Na czym polega optymalizacja *register blocking*? Podaj przykład optymalizacji i jej wpływ na wydajność

18. *Na czym polegają optymalizacje: software prefetching i software pipelining? Podaj przykłady optymalizacji i ich wpływ na wydajność*

19. Na czym polega optymalizacja *cache blocking*? Podaj przykład optymalizacji i jej wpływ na wydajność

B

1. W trakcie sekwencyjnego (jednowątkowego) wykonania fragmentu kodu, w którym o czasie wykonania decydował tylko czas realizacji ..... operacji zmiennoprzecinkowych, profiler podał czas wykonania jako czas ..... taktów rzeczywistych i ..... taktów referencyjnych. Jaką wydajność w [flop/takt] uzyskał rdzeń mikroprocesora? Zakładając nominalną (zawartą np. w pliku */proc/*cpuinfo) częstotliwość pracy równą ..... GHz, oblicz jaka była wydajność rdzenia w Gflop/s przy wykonaniu kodu.

2. Skonstruuj diagram *roofline* dla platformy sprzętowej charakteryzującej się maksymalną wydajnością potoków przetwarzania ..... Gflop/s i przepustowością pamięci ..... GB/s. Wskaż na diagramie punkt oznaczany jako *ridge point*, związany z parametrem określanym jako *machine balance*. Czym jest ten parametr? Jaka jest jego wartość dla przeciętnych współczesnych mikroprocesorów (a także ich pojedynczych rdzeni)?

3. Skonstruuj diagram *roofline* dla platformy sprzętowej charakteryzującej się maksymalną wydajnością potoków przetwarzania ..... Gflop/s i przepustowością pamięci ..... GB/s. Zaznacz na diagramie linie i ewentualne punkty związane z wydajnością dwóch programów: jednego, którego wydajność jest ograniczana przez maksymalną wydajność pobierania danych z pamięci i drugiego, którego wydajność jest ograniczana przez maksymalną wydajność potoków przetwarzania.

4.
5.
6. Program podczas wykonania realizuje ..... razy pętle odpowiadającą kodowi asemblera:

a)

```
..B2.2:
        vfmadd231pd %ymm10, %ymm6, %ymm7
        vfmadd231pd %ymm10, %ymm6, %ymm8
        vmovupd   %ymm9, 320(%rsp,%rax,8)
        vmovupd   %ymm10, 448(%rsp,%rax,8)
        addq      $4, %rax
        cmpq      $16, %rax
        jb        ..B2.2
```

b)

```
.L4:
        movl    -4(%rbp), %eax
        imull   -44(%rbp), %eax
        addl    %edx, %eax
        leaq    0(,%rax,8), %rdx
        movq    -24(%rbp), %rax
        addq    %rdx, %rax
        movsd   (%rax), %xmm1
        mulsd   %xmm1, %xmm0
        addl    $1, -8(%rbp)
.L3:
        movl    -8(%rbp), %eax
        cmpl    -44(%rbp), %eax
        jl      .L4
```

c)

```
.L5:
        movsd   (%rdi,%rax,8), %xmm0
        mulsd   (%rsi,%rax,8), %xmm0
        addq    $1, %rax
        cmpl    %eax, %ecx
        addsd   %xmm0, %xmm1
        jg      .L5
```

d)

```
.L7:
        vmovsd  (%rcx,%rax), %xmm0
        vfmadd213sd     (%rdx,%rax), %xmm1, %xmm0
        vmovsd  %xmm0, (%rdx,%rax)
        addq    $8, %rax
        cmpq    $288, %rax
        jne     .L7
```

e)

```
.L23:
        vmovupd (%rdx,%rax), %ymm1
        vmovupd (%rcx,%rax), %ymm2
        vbroadcastsd    (%rsi), %ymm0
        vfmadd132pd     %ymm2, %ymm1, %ymm0
        vmovupd %ymm0, (%rdx,%rax)
        addq    $32, %rax
        cmpq    $864, %rax
        jne     .L23
```

f)

```
.L6:
        vmovsd  (%rdx,%rax), %xmm3
        vmovsd  (%rsi), %xmm0
        vfmadd132sd     (%rcx,%rax), %xmm3, %xmm0
        vmovsd  %xmm0, (%rdx,%rax)
        addq    $8, %rax
        cmpq    $864, %rax
        jne     .L6
```

g)

```
..B1.2:
        lea       128(%rsp), %rax
```

```
        movsd     (%rax), %xmm1
        mulsd     %xmm1, %xmm0
        incl      %eax
        addsd     %xmm2, %xmm0
        cmpl      $1000000000, %eax
        jl        ..B1.2
```

h)

```
        lub dowolnemu innemu ...
```

Podaj ile pojedynczych operacji arytmetycznych (dodawań i mnożeń) wykonuje rdzeń procesora realizując pętlę? Ile danych jest przesyłanych do procesora z pamięci (dowolnego poziomu)?

7.
8.
9.
10. Analiza kodu źródłowego, asemblera, zliczania zdarzeń sprzętowych i symulacji za pomocą programów profilujących doprowadziła do wniosku, że w trakcie wykonania programu:
    - procesor wykonuje ..... skalarnych rozkazów zmiennoprzecinkowych fadd, fmul, fma na zmiennych 64-bitowych oraz
    - procesor wykonuje ..... wektorowych rozkazów zmiennoprzecinkowych fadd, fmul, fma na wektorach 256-bitowych oraz
    - procesor wykonuje ..... rozkazów skalarnych pobrania danych podwójnej precyzji do rejestrów 64-bitowych
    - procesor wykonuje ..... rozkazów wektorowych pobrania danych do rejestrów 256-bitowych
    - występuje ..... chybień w pamięci L1 (zliczanych np. przez zdarzenie L1D.REPLACEMENT) powodujących pobranie danych z pamięci L2 (np. dla pamięci podręcznych typu *inclusive*) - co oznacza transfer ..... GB danych z pamięci L2
    - występuje ..... chybień w pamięci L1 (zliczanych np. przez zdarzenie L1D.REPLACEMENT) powodujących pobranie danych z pamięci L3 (np. dla pamięci podręcznych typu ex*clusive*) - co oznacza transfer ..... GB danych z pamięci L3
    - występuje ..... chybień w pamięci L2 (zliczanych np. przez zdarzenie L2_LINES_IN.ALL) powodujących pobranie danych z pamięci L3 - co oznacza transfer ..... GB danych z pamięci L3
    - występuje ..... chybień w pamięci L3 (zliczanych np. przez zdarzenie MEM_LOAD_UOPS_L3_MISS_RETIRED.LOCAL_DRAM) powodujących pobranie danych z pamięci DRAM - co oznacza transfer ..... GB danych z pamięci DRAM

    Zakładając, że wszystkie inne operacje poza wymienionymi powyżej są wykonywane w tle i nie wpływają na czas wykonania programu, oblicz jaki jest minimalny czas wykonania na platformie charakteryzującej się maksymalną wydajnością potoków przetwarzania ..... Gflop/s i przepustowością pamięci: ..... GB/s dla pamięci DRAM, ... GB/s dla pamięci L3, ... GB/s dla pamięci L2 i ..... GB/s dla pamięci L1.

11. *Przeprowadź analizę skalowalności w sensie słabym dla implementacji z przesyłaniem komunikatów algorytmu:*
    - *obliczania iloczynu skalarnego dwóch wektorów*
    - *obliczania iloczynu macierz-wektor*
        - *dla macierzy gęstych*
        - *dla macierzy pasmowych*

12. Wyprowadź wzory na czas realizacji operacji komunikacji grupowej rozgłaszania (*broadcast*) / redukcji (*reduction*) / rozpraszania (*scatter*) / zbierania (*gather*) dla topologii pierścienia / torusa 2D / hiperkostki

---

C

1. Podaj przykłady rozkazów procesora: operacji arytmetycznych, logicznych, skoków – warunkowych i bezwarunkowych, dostępów do pamięci; użyj jednej ze stosowanych notacji i przykładowego procesora (istotna jest idea notacji i rozkazu, nie ścisła poprawność składniowa)

2. Podaj przykłady skalarnych i wektorowych rozkazów procesora – operacji arytmetycznych i dostępów do pamięci; użyj jednej ze stosowanych notacji i przykładowego procesora (istotna jest idea notacji i rozkazu, nie ścisła poprawność składniowa)

3. Podaj przykłady rozkazów dostępu do pamięci ze złożonym adresowaniem, wyjaśnij rolę poszczególnych parametrów

4. Jaka jest zasada przetwarzania potokowego?

5. Porównaj pod kątem wydajności przetwarzanie rozkazów przez procesor (rdzeń): bez potokowości i z potokowością

6. Czym jest parametr CPI, a czym IPC? Jak wartość maksymalna tego ostatniego zależy od architektury procesora?

7. Wymień i krótko scharakteryzuj kilka zaawansowanych cech współczesnych procesorów (rdzeni) zwiększających wydajność klasycznego procesora potokowego.

8. Jak obliczać wydajność maksymalną potoków przetwarzania procesora? Jakie są ograniczenia praktycznego użycia ogólnych wartości IPC i CPI? Jak praktycznie oblicza się maksymalną wydajność rdzenia, mikroprocesora, komputera dla operacji zmiennoprzecinkowych?

9. Czym różni się opóźnienie przy wykonywaniu rozkazów przez procesor od (maksymalnej) przepustowości? Jak maksymalizować rzeczywistą przepustowość (średnie IPC) przy wykonaniu programu?

10. Jak obliczyć współczynniki CPI i IPC dla konkretnego wykonania programu? Jakich narzędzi można do tego użyć? Jakie są ograniczenia i trudności z interpretacją dla pomiaru CPI i IPC standardowych programów?

11. Jak skoki, warunkowe i bezwarunkowe, zaburzają przetwarzanie potokowe? W jaki sposób można uniknąć opóźnień przetwarzania potokowego w takiej sytuacji?

12. Czym są zależności danych między rozkazami wykonywanymi przez procesor? W jaki sposób zaburzają przetwarzanie potokowe? W jaki sposób można uniknąć opóźnień przetwarzania potokowego w takiej sytuacji?

13. Jak wygląda faza pobierania i dekodowania oraz przekazywania do wykonania rozkazów przez współczesne procesory?

14. W jaki sposób procesor może przeprowadzac przewidywanie skoków (rozgałęzień)?

15. Przedstaw i objaśnij tzw. równanie wydajności (*performance equation*). Co oznaczają i jak można optymalizować wartości czynników występujących we wzorze?

16. Czym jest blok podstawowy w kodzie asemblera? Jak wyszukiwać bloki podstawowe? Jaka jest rola bloków podstawowych w procesie optymalizacji?

17. Jak działa pamięć wirtualna ze stronicowaniem?

18. Opisz proces translacji adresów wirtualnych na fizyczne. Czym jest tablica stron (*page table*)?

19. Co to jest TLB (*translation lookaside buffer*)? Kiedy następuje intensywne użycie TLB?

20. Czym jest mniejszy błąd strony (*minor page fault*), a czym większy błąd strony (*major page fault*)

21. Czym jest szamotanie (*thrashing*)? Jak można go unikać?

22. Jaki jest w przybliżeniu czas wykonania pojedynczej operacji arytmetycznej przez procesor, a jaki czas pobrania pojedynczej zmiennej z pamięci DRAM, zakładając izolowane operacje (czyli czasy typowe dla opóźnienia, *latency*)

23. Jaki jest w przybliżeniu czas wykonania pojedynczej operacji arytmetycznej przez procesor, a jaki czas pobrania pojedynczej zmiennej z pamięci DRAM, zakładając długie sekwencje optymalnie wykonywanych operacji (czyli czasy typowe dla przepustowości, *throughput*)

24. Jak działa pamięć podręczna (*cache memory*)? Omów na przykładzie jednopoziomowej pamięci podręcznej.

25. Skąd bierze się pozytywny wpływ istnienia pamięci podręcznej na wydajność pracy procesora? Jakie są typy lokalności odniesień?

26. Jak działa pamięć podręczna bezpośrednio odwzorowana (*direct mapped*) w odróżnieniu od pamięci w pełni skojarzeniowej (*fully associative*)

27. Co to jest drożność pamięci podręcznej? Podaj schemat działania dwudrożnej pamięci sekcyjno-skojarzeniowej (*2-way set associative*).

28. Kiedy występuje chybienie wymuszone (*compulsory miss*) w pamięci podręcznej?

29. Kiedy występuje chybienie pojemnościowe (*capacity miss*) w pamięci podręcznej?

30. Kiedy występuje chybienie konfliktowe (*conflict miss*) w pamięci podręcznej?

31. Jakie są mechanizmy ukrywania opóźnienia przy dostępach do pamięci? Jak pisać kod maksymalizujący wydajność pamięci?

32. Jakie zasoby są wspólne, a jakie odrębne dla wątków tego samego procesu?

33. Co to jest przełączanie kontekstu? W jaki sposób opóźnia działanie programu?

34. Czym jest sprzętowa jednoczesna wielowątkowość (*simultaneous multithreading*)? Jak reprezentowana jest na poziomie systemu operacyjnego? Jak wpływa na wydajność programów?

35. Czym różni się organizacja UMA i NUMA dostępu do pamięci? Jaki ma wpływ na wydajność?

36. Przedstaw strategie przypisywania wątków do rdzeni (procesorów logicznych dla SMT)

37. Omów strategie *write-through*, *write-back*, *write-allocate*, *no-write-allocate* zapisu danych do pamięci.

38. Scharakteryzuj protokoły utrzymywania zgodności pamięci podręcznej (*cache coherence protocols*) z unieważnianiem (np. protokół MESI)

39. Kiedy występuje chybienie spójnościowe (*coherency miss*) w pamięci podręcznej?

40. Czym jest zjawisko fałszywego współdzielenia (*false sharing*) i jak wpływa na wydajność programów?

D

1. Co to jest, do czego służy i jak można uzyskać profil wykonania kodu?

2. Programy profilujące (profilery) – zasady działania, przykłady

3. Czas wykonania – zewnętrzny, CPU, użytkownika, systemowy, narzędzia systemowe zwracające czas wykonania – dla programów jednowątkowych i wielowątkowych

4. Czym są liczniki sprzętowe i do czego można je wykorzystać? Podaj przykłady zdarzeń sprzętowych (*hardware events*)

5. Narzędzia (systemowe i inne) korzystania z liczników sprzętowych

6. Na czym polega strategia optymalizacji przez usuwanie wąskich gardeł (*bottlenecks*)?

7. Jakie są typowe kroki podejmowane przy optymalizacji kodu ze względu na wydajność?

8. Co ogranicza wydajność w pojedynczym węźle obliczeniowym?

9. Co to jest diagram *roofline* i do czego służy? W jaki sposób uzyskiwane są proste na diagramie *roofline*?

10. Jak uzyskać poziomą linię na eksperymentalnym diagramie *roofline?* (czyli jak zaprojektować benchmark do maksymalnego wykorzystania mocy potoków przetwarzania)? Jakie warianty można rozważyć?

11. Jak uzyskać nachyloną linię na diagramie *roofline*? (czyli jak zaprojektować benchmark do pełnego wykorzystania możliwości układu pamięci i jak wykorzystać jego wyniki)

12. Jak obliczyć współczynnik s_{pm} dla programu? Uwzględnij warianty dla pamięci DRAM i pamięci podręcznych różnych poziomów.

13. Jak sprawdzić, korzystając z diagramu *roofline*, czy optymalizowany program dopuszcza jeszcze znaczące zyski wydajności?

14. Jaki jest wpływ zwiększania lokalności odniesień w kodzie na współczynnik intensywności arytmetycznej s_{pm}?

15. Jakie optymalizacje mogą zwiększyć współczynnik intensywności arytmetycznej s_{pm} i dlaczego?

16. Jak wykorzystać diagram *roofline* do optymalizacji algorytmu?

17. Kiedy mówimy, że wydajność programu jest ograniczana przez wydajność pamięci?

18. Kiedy mówimy, że wydajność programu jest ograniczana przez wydajność (potoków przetwarzania) procesora?

19. Jakie jest znaczenie punktu przecięcia linii na diagramie *roofline*, odpowiadającego tzw. równowadze sprzętu (*machine balance*)?

20. Dlaczego programy mogą osiągać wydajności znacznie niższe od maksymalnej wydajności potoków przetwarzania, przy optymalnym wykorzystaniu dostępnego sprzętu?

21. Jakie dwa podstawowe czynniki (dwie podstawowe cechy wydajnościowe) wpływają na ostateczną bezwzględną wydajność programów równoległych? Podaj odpowiednie wzory dla wydajności wyrażanej w Gflop/s.

22. Jak w przybliżeniu można szacować czas przesłania pojedynczego komunikatu o *m* bajtach, pomiędzy dwoma węzłami sieci komputerowej?

23. Jak należy przeprowadzać komunikacje grupową, w celu minimalizacji czasu operacji? Podaj przykład abstrakcyjny lub dla konkretnej architektury sieci

24. Scharakteryzuj topologię *hiperkostki* sieci komputerowych łączących węzły obliczeniowe?

25. Scharakteryzuj topologię torusa sieci komputerowych łączących węzły obliczeniowe?

26. Co to jest przyspieszenie obliczeń i efektywność zrównoleglenia?

27. Czym różni się badanie skalowalności w sensie słabym od badania skalowalności w sensie silnym?

28. Jak tworzy się wykres przyspieszenia przeskalowanego (*scaled speed-up*)?

29. Kiedy mówimy, że program jest skalowalny w sensie słabym?

30. Jaki jest warunek dotyczący czasu wykonania równoważny liniowej skalowalności w sensie słabym?

31. Omów pojecie narzutu czasowego obliczeń równoległych. Jak powinien zachowywać się narzut, żeby obliczenia były skalowalne w sensie silnym, a co wystarcza do skalowalności w sensie słabym?

32. Jakie czynniki wpływają na wydajność programów równoległych (stanowią narzut wykonania równoległego)?