2025年度入学試験

東京大学大学院情報理工学系研究科 コンピュータ科学専攻 試験問題

専門科目（問題1, 2, 3, 4）

2024年8月19日

13:00 – 15:30

注意事項

(1) 試験開始の合図があるまで, この問題冊子を開けないこと.

(2) 4題すべてに答えよ. 問題ごとに指定された解答用紙を使用すること.

(3) 解答用紙および問題冊子は持ち帰らないこと.

Specialized Subjects (Problems 1, 2, 3, and 4)

13:00 – 15:30, August 19, 2024

AY 2025 Entrance Examination

Department of Computer Science, Graduate School of Information Science and Technology

The University of Tokyo

Notice:

(1) Do not open this problem booklet until the start of the examination is announced.

(2) Answer the following 4 problems. Use the designated answer sheet for each problem.

(3) Do not take the problem booklet or any answer sheet out of the examination room.

下欄に受験番号を記入すること.

Write your examinee number in the box below.

受験番号	No.

問題 1

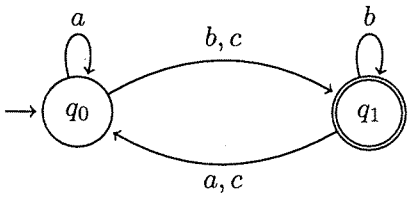
$\Sigma = \{a, b, c\}$ とする． Σ 上の言語 $L \subseteq \Sigma^*$ に対して，言語 $\mathcal{H}(L) \subseteq \Sigma^*$ を以下によって定義する．

$$\mathcal{H}(L) = \{w \in \Sigma^* \mid ww \in L\}.$$

例えば，$L_1 = \{aa, abc, abab, baab, cca\}$ ならば，$\mathcal{H}(L_1) = \{a, ab\}$ である．
以下の問いに答えよ．

(1) L_2 を正規表現 $a(a+b)^*c(a+b)^*bc$ で表される言語とする． $\mathcal{H}(L_2)$ を正規表現として表せ．

(2) 以下の有限オートマトンによって受理される言語を L_3 とする．ただし，初期状態は q_0 であり，受理状態の集合は $\{q_1\}$ である． $\mathcal{H}(L_3)$ を受理する状態数最小の決定性有限オートマトンを構成せよ．



(3) 以下の命題 1 が真であるか否かを答え，真であればその証明を，そうでなければ簡単な説明とともに反例を示せ．

> **命題 1**: Σ 上のすべての正規言語 $L \subseteq \Sigma^*$ について，$\mathcal{H}(L)$ も正規言語である．

(4) 以下の命題 2 が真であるか否かを答え，真であればその証明を，そうでなければ簡単な説明とともに反例を示せ．

> **命題 2**: Σ 上のすべての文脈自由言語 $L \subseteq \Sigma^*$ について，$\mathcal{H}(L)$ も文脈自由言語である．

Problem 1

Let $\Sigma = \{a, b, c\}$. For a language $L \subseteq \Sigma^*$ over Σ, the language $\mathcal{H}(L) \subseteq \Sigma^*$ is defined by:

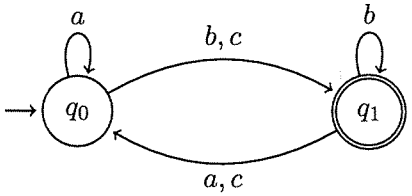
$$\mathcal{H}(L) = \{w \in \Sigma^* \mid ww \in L\}.$$

For example, if $L_1 = \{aa, abc, abab, baab, cca\}$, then $\mathcal{H}(L_1) = \{a, ab\}$.

Answer the following questions.

(1) Let L_2 be the language expressed by the regular expression $a(a + b)^*c(a + b)^*bc$. Express $\mathcal{H}(L_2)$ as a regular expression.

(2) Let L_3 be the language accepted by the finite automaton given below. Here, q_0 is the initial state and $\{q_1\}$ is the set of accepting states. Construct a deterministic finite automaton with the minimum number of states which accepts $\mathcal{H}(L_3)$.



(3) Answer whether Proposition 1 given below is true or not. If it is true, then give a proof; otherwise, give a counterexample with a brief explanation.

> **Proposition 1**: For every regular language $L \subseteq \Sigma^*$ over Σ, $\mathcal{H}(L)$ is also a regular language.

(4) Answer whether Proposition 2 given below is true or not. If it is true, then give a proof; otherwise, give a counterexample with a brief explanation.

> **Proposition 2**: For every context-free language $L \subseteq \Sigma^*$ over Σ, $\mathcal{H}(L)$ is also a context-free language.

問題 2

毎クロックサイクル最大1命令を発行する5ステージのパイプラインプロセッサPを考える．Pは，x0からx7までの8個のレジスタを持ち，ロード命令とストア命令のデータ幅は4バイトとする．Pは命令を遅延なしで読み出しできる．Pはデータキャッシュを搭載し，キャッシュミス時は8クロックサイクルの間ストールし，キャッシュヒット時はストールしない．また，ロード命令の直後の命令がロード結果を利用する場合には，1クロックサイクルの間ストールする．その他の理由ではPはストールせず，プリフェッチ機構はないものとする．Pは分岐予測に基づき分岐命令の後続2命令を投機的にフェッチする．分岐予測ミス時は，投機的にフェッチした2命令をフラッシュした後，分岐先の正しい命令をフェッチする．

以下の問いに答えよ．

(1) あるプログラムSをP上で実行すると，実行命令数は 1,000，要したクロックサイクル数は 1,154 であった．Sは，20%のロード命令，15%のストア命令，10%の分岐命令，55%の算術論理演算命令で構成されており，ロード命令の40%は直後にその結果を利用する命令が続く．Sの実行ではキャッシュミスは発生しないものとする．分岐予測ミス率を小数第2位まで求めよ．

(2) 下のプログラムをP上で実行すると，1,796 クロックサイクル要した．各命令の動作はプログラム中のコメント（#以降の記述）の通りとし，memory[addr] は addr 番地へのメモリアクセスを表す．addi と add は算術論理演算命令，lw はロード命令，sw はストア命令，blt は分岐命令である．Pのデータキャッシュは，ダイレクトマップ方式で，データ容量は4096バイトとする．データキャッシュの初期状態は空とする．このプログラムの実行では分岐予測ミスは発生しないものとする．x1, x2, x3 の初期値はそれぞれ，0, 1024, 0 とする．Pのキャッシュラインサイズを求めよ．

```
Loop: lw x4, 0 (x1)      # x4 <- memory[x1 + 0]
      add x3, x3, x4     # x3 <- x3 + x4
      sw x3, 0 (x1)      # memory[x1 + 0] <- x3
      addi x1, x1, 4     # x1 <- x1 + 4
      blt x1, x2, Loop   # if x1 < x2, goto Loop
```

(3) 問い (2) のプログラム中のすべての命令に対応するプロセッサの16ビットの命令表現として，以下の表に示すFormat AとFormat Bを考える．X[Y] はXのYビット目を示し，X[H:L] はXのHビット目からLビット目までのビット列を示す．{X, Y} は，XとYのビット連結を示す．rs1 と rs2 はソースオペランド，rd はデスティネーションオペランド，imm は即値をそれぞれ示す．ただし，指定されていない即値のビットは0とする．Format AとFormat Bのどちらがオペランドや即値のデコードにより多くの回路資源を消費するかを理由と共に答えよ．

Format A

[15:12]	[11:9]	[8:6]	[5:3]	[2:0]	
0x0	rs2	rs1	rd	0x0	add
imm[6:0]		rs1	rd	0x1	addi
imm[6:0]		rs1	rd	0x2	lw
imm[6:0]		rs1	rs2	0x3	sw
imm[7:1]		rs1	rs2	0x4	blt

Format B

[15:12]	[11:9]	[8:6]	[5:3]	[2:0]	
0x0	rs2	rs1	rd	0x0	add
imm[6:0]		rs1	rd	0x1	addi
imm[6:0]		rs1	rd	0x2	lw
imm[6:3]	rs2	rs1	imm[2:0]	0x3	sw
{imm[7],imm[5:3]}	rs2	rs1	{imm[2:1],imm[6]}	0x4	blt

Problem 2

Consider a 5-stage pipeline processor P that issues up to one instruction per clock cycle. P has eight registers from x0 to x7, and the data widths of the load/store instructions are 4 bytes. P can read instructions without delay. P has a data cache. P stalls for eight clock cycles for each cache miss, and there is no stall for cache hits. P stalls for one clock cycle when the next instruction of a load instruction uses the load result. P will not stall for any other reasons and has no prefetching mechanism. P fetches the next two instructions after a branch instruction speculatively based on the branch prediction. In case of a branch misprediction, P flushes the speculatively fetched two instructions and then fetches a correct instruction from the branch target.

Answer the following questions.

(1) Suppose that, for the execution of some program S on P, 1,000 instructions were executed, and it took 1,154 clock cycles. S consists of 20% load instructions, 15% store instructions, 10% branch instructions, and 55% arithmetic and logic instructions. Assume that 40% of the load instructions are immediately followed by an instruction that uses the result. Assume that no cache miss occurs for the execution of S. Calculate the branch misprediction rate up to two places of decimals.

(2) Suppose that the execution of the program below on P took 1,796 clock cycles. The behavior of each instruction is described as a comment (the description after #) in the program, where memory[addr] represents a memory access to the address addr. addi and add are arithmetic and logic instructions, lw is a load instruction, sw is a store instruction, and blt is a branch instruction. The data cache of P is direct-mapped and stores 4096 bytes of data. The initial state of the data cache is empty. Assume that no branch misprediction occurs for the execution of the program. The initial values of x1, x2, and x3 are 0, 1024, and 0, respectively. Calculate the cache line size of P.

```
Loop: lw x4, 0 (x1)      # x4 <- memory[x1 + 0]
      add x3, x3, x4     # x3 <- x3 + x4
      sw x3, 0 (x1)      # memory[x1 + 0] <- x3
      addi x1, x1, 4     # x1 <- x1 + 4
      blt x1, x2, Loop   # if x1 < x2, goto Loop
```

(3) Consider Format A and Format B in the table below as the 16-bit instruction representation for a processor that supports all instructions in the program of Question (2). X[Y] represents the Y-th bit of X, and X[H:L] represents bits from the H-th bit to the L-th bit of X. {X, Y} represents a bit concatenation of X and Y. rs1 and rs2 represent source operands, rd represents a destination operand, and imm represents an immediate value, respectively. Assume that unspecified bits of the immediate value are 0. Answer which of Format A and Format B consumes more circuit resources for decoding the operands and immediate value, and explain why.

Format A

[15:12]	[11:9]	[8:6]	[5:3]	[2:0]	
0x0	rs2	rs1	rd	0x0	add
imm[6:0]		rs1	rd	0x1	addi
imm[6:0]		rs1	rd	0x2	lw
imm[6:0]		rs1	rs2	0x3	sw
imm[7:1]		rs1	rs2	0x4	blt

Format B

[15:12]	[11:9]	[8:6]	[5:3]	[2:0]	
0x0	rs2	rs1	rd	0x0	add
imm[6:0]		rs1	rd	0x1	addi
imm[6:0]		rs1	rd	0x2	lw
imm[6:3]	rs2	rs1	imm[2:0]	0x3	sw
{imm[7],imm[5:3]}	rs2	rs1	{imm[2:1],imm[6]}	0x4	blt

問題 3

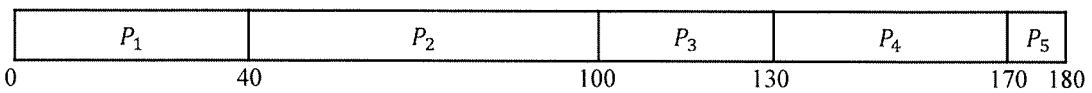
オペレーティングシステムにおける，プロセス $P_i\,(i=1,2,3,4,5)$ のスケジューリングを考える．P_1, P_2, P_3, P_4, P_5 の到着時刻はそれぞれ 0, 5, 10, 15, 20，処理時間は 40, 60, 30, 40, 10 である．次の 4 つのスケジューリングアルゴリズムを考える．(a) first come, first serve (FCFS), (b) (non-preemptive) shortest job first (SJF), (c) shortest remaining time first (SRTF), (d) round robin (RR) (time quantum は 30)．ただし，一度に実行できるプロセスは一つだけで，言及のないコストは無視できるものとする．

また，バイナリセマフォ $S_j\,(j=1,2)$ を取得，解放するための操作をそれぞれ $wait(S_j)$, $signal(S_j)$ とする．操作 $wait(S_j)$ は，S_j を取得できなかったら，実行したプロセスを S_j の待ちキューの末尾に入れる．操作 $signal(S_j)$ は，S_j を解放したあと，S_j の待ちキューが空でなければ，その先頭のプロセスを実行可能キューに移動する．

なお，プロセスの待ち時間は，実行可能キューまたは $S_j\,(j=1,2)$ の待ちキューで待つ時間の合計とする．プロセスの平均待ち時間とは，各プロセス $P_i\,(i=1,2,3,4,5)$ の待ち時間の平均のことである．

(a)〜(d) のそれぞれのスケジューリングアルゴリズムについて，以下の問いに答えよ．ただし，以下の問い (1), (2) では，プロセスはセマフォ操作はおこなわないものとする．

(1) $P_i\,(i=1,2,3,4,5)$ がスケジュールされる時刻を，以下のようなガントチャートの形で描け．



(2) プロセスの平均待ち時間を求めよ．

(3) P_1, P_3, P_5 は起動時に $wait(S_1)$，終了時に $signal(S_1)$ をそれぞれ実行し，P_2, P_4 はセマフォ操作をおこなわないとする．このときのプロセスの平均待ち時間を求めよ．

(4) P_i が以下のようにセマフォ操作をおこなう場合を考える．

- P_1 と P_2 は，起動時に $wait(S_1)$，実行時間が 15 経過後に $wait(S_2)$，終了時に $signal(S_2)$ 及び $signal(S_1)$ をこの順に実行する．
- P_3 と P_4 は，起動時に $wait(S_2)$，実行時間が 15 経過後に $wait(S_1)$，終了時に $signal(S_1)$ 及び $signal(S_2)$ をこの順に実行する．
- P_5 はセマフォ操作をおこなわない．

このとき，デッドロックが発生する場合はその発生時刻を求めよ．デッドロックが発生しない場合はプロセスの平均待ち時間を求めよ．

Problem 3

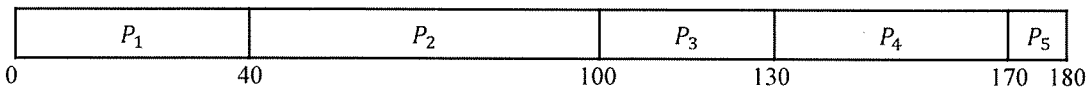
Consider the scheduling of processes $P_i\,(i = 1, 2, 3, 4, 5)$ in an operating system. The arrival times of P_1, P_2, P_3, P_4 and P_5 are 0, 5, 10, 15 and 20, and processing times are 40, 60, 30, 40 and 10, respectively. We consider the following four scheduling algorithms: (a) first come, first serve (FCFS), (b) (non-preemptive) shortest job first (SJF), (c) shortest remaining time first (SRTF), and (d) round robin (RR) (time quantum is 30). Here, we assume that only one process can run at a time and unmentioned costs can be ignored.

Also, the operations to acquire and release binary semaphores $S_j\,(j = 1, 2)$ are denoted as $wait(S_j)$ and $signal(S_j)$, respectively. If the operation $wait(S_j)$ cannot acquire S_j, it puts the executing process at the end of the waiting queue for S_j. The operation $signal(S_j)$ releases S_j, and if the waiting queue for S_j is non-empty, then the first process in the waiting queue is moved to the ready queue.

Here, the waiting time of a process is the total amount of time the process spends waiting in the ready queue or the waiting queue for $S_j\,(j = 1, 2)$. The average waiting time of the processes is the average of the waiting time of each process $P_i\,(i = 1, 2, 3, 4, 5)$.

For each of the scheduling algorithms (a) to (d), answer the following questions. In questions (1) and (2), assume that the processes do not perform any semaphore operations.

(1) Draw the time P_i $(i = 1, 2, 3, 4, 5)$ is scheduled in the form of a Gantt chart as shown below.



(2) Calculate the average waiting time of the processes.

(3) Suppose that P_1, P_3, and P_5 perform $wait(S_1)$ at startup and $signal(S_1)$ at termination, respectively, and P_2 and P_4 do not perform any semaphore operations. Calculate the average waiting time of the processes in this case.

(4) Consider the case where P_i performs semaphore operations as follows.

- P_1 and P_2 perform $wait(S_1)$ at startup, $wait(S_2)$ after a runtime of 15 elapsed, and $signal(S_2)$ and $signal(S_1)$ at termination in this order.
- P_3 and P_4 perform $wait(S_2)$ at startup, $wait(S_1)$ after a runtime of 15 elapsed, and $signal(S_1)$ and $signal(S_2)$ at termination in this order.
- P_5 does not perform any semaphore operations.

In this case, if a deadlock occurs, find the time when the deadlock occurs. If no deadlock occurs, calculate the average waiting time of the processes.

問題 4

以下では，自己ループ（同一の点を結ぶ辺）も多重辺（同一の2点を結ぶ2本以上の辺）も含まない無向グラフ $G = (V_G, E_G)$ を考える．正の整数 k について，G が k **連結**であるとは，$|V_G| > k$ であり，かつ G からどのように $k-1$ 点を削除しても，非連結にならないことをいう．また，G の部分グラフ C が**サイクル**であるとは，C が2連結であり，C のすべての頂点が C において次数2であることをいう．次の定理 (I) および (II) を解答に用いてよい．

(I) 任意の無向グラフ G，隣接しない（すなわち，辺で直接結ばれていない）異なる2点 $a, b \in V_G$ および正の整数 k について，以下のいずれかが成立する（Menger の定理）．

  (i) a, b 間を結ぶ内点素なパスが k 本存在する（2つのパスが**内点素**であるとは，パスの両端点以外に点を共有しないことである）．

  (ii) a, b 以外の $k-1$ 点以下を G から削除することによって，a, b 間のパスが存在しないようにできる．

(II) 無向グラフ G，隣接しない異なる2点 $a, b \in V_G$，および正の整数 k を入力とし，上記 (I) の2つの条件 (i), (ii) のいずれが成り立つかを判定するための $|V_G|, |E_G|, k$ に関する多項式時間アルゴリズムが存在する．

以下の問いに答えよ．

(1) 無向グラフ G および異なる2点 $a, b \in V_G$ を入力とし，a, b の両方を含むサイクルが G に存在するか否かを判定する，$|V_G|, |E_G|$ に関する多項式時間アルゴリズムを与えよ．ただし，上の (II) のアルゴリズムを用いてよい．

(2) 任意の正の整数 k について，無向グラフ G が k 連結であれば，いかなる k 点集合 $A \subseteq V_G$ についても以下が成立することを証明せよ．

> 「新たな点 u を G に追加し，A のすべての点 v について辺 (u, v) を追加して得られるグラフを G' とする．このとき，G' も k 連結である．」

(3) 無向グラフ G が3連結ならば，任意の3点 $a, b, c \in V_G$ について，a, b, c をすべて含むサイクルが G に存在することを証明せよ．

(4) 任意の整数 $k \geq 2$ について，無向グラフ G が k 連結ならば，任意の k 点集合 $A \subseteq V_G$ について，A のすべての点を含むサイクルが G に存在することを証明せよ．

Problem 4

Below we consider an undirected graph $G = (V_G, E_G)$ with no self-loops (edges joining the same vertex) nor multi-edges (two or more edges joining the same two vertices). For a positive integer k, we say that G is *k-connected* if $|V_G| > k$ and it remains connected after removing any $k-1$ vertices. A subgraph C of G is called a *cycle* if C is 2-connected and every vertex of C has degree 2 in C. You may use the following theorems (I) and (II) in your answers.

(I) For any undirected graph G, any two non-adjacent (i.e., not directly connected by an edge), distinct vertices $a, b \in V_G$, and any positive integer k, one of the following conditions holds (Menger's theorem).

  (i) There exist k internally vertex disjoint paths between a and b (two paths are *internally vertex disjoint* if they do not share any vertex, except the two endpoints).

  (ii) By removing at most $k-1$ vertices other than a, b from G, one can ensure that there exists no path between a and b.

(II) There exists a polynomial time algorithm in $|V_G|, |E_G|$, and k which, given an undirected graph G, two non-adjacent, distinct vertices $a, b \in V_G$, and a positive integer k as input, decides which of the two conditions (i) and (ii) of (I) above holds.

Answer the following questions.

(1) Give a polynomial time algorithm in $|V_G|$ and $|E_G|$ which, given an undirected graph G and two distinct vertices $a, b \in V_G$ as input, decides whether there exists a cycle that contains both a and b in G. Here, you may use the algorithm of (II) above.

(2) Prove that, for any positive integer k, if an undirected graph G is k-connected, then the following holds for any set $A \subseteq V_G$ consisting of k vertices.

> "Let G' be the graph obtained from G by adding a new vertex u, and adding an edge (u, v) for every vertex v of A. Then, G' is also k-connected."

(3) Prove that if an undirected graph G is 3-connected, then for any three vertices $a, b, c \in V_G$, G has a cycle containing all of a, b, and c.

(4) Prove that, for any integer $k \geq 2$, if an undirected graph G is k-connected, then for any set $A \subseteq V_G$ consisting of k vertices, G has a cycle containing all the vertices in A.