2026 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）

2025 年 8 月 20 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 20, 2025

AY 2026 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.

余白 (blank page)

計算などに使ってもよいが，切り離さないこと． Usable for memos; do not detach.

余白 (blank page)

計算などに使ってもよいが，切り離さないこと． Usable for memos; do not detach.

問題 1

$\Sigma = \{\mathtt{a}, \mathtt{b}\}$ とする．Σ 上の語 $w_1, w_2 \in \Sigma^*$ に対して，$w_1 \odot w_2 \subseteq \Sigma^*$ を以下によって定義する．

$$w_1 \odot w_2 = \{v_1 \cdots v_{k+1} \mid \\ k \geq 0 \wedge a_1, \ldots, a_k \in \Sigma \wedge v_1, \ldots, v_{k+1} \in \Sigma^* \wedge w_1 = v_1 a_1 v_2 \cdots v_k a_k v_{k+1} \wedge w_2 = a_1 \cdots a_k\}.$$

すなわち，$w_1 \odot w_2$ は w_1 から部分列 w_2 を取り除いて得られる語の集合を表す．例えば，$\mathtt{abababa} \odot \mathtt{bb} = \{\mathtt{aaaba}, \mathtt{aabaa}, \mathtt{abaaa}\}$ である．

さらに，Σ 上の言語 $L_1, L_2 \subseteq \Sigma^*$ に対して $L_1 \ominus L_2 \subseteq \Sigma^*$ を以下によって定義する．

$$L_1 \ominus L_2 = \bigcup_{w_1 \in L_1, w_2 \in L_2} w_1 \odot w_2.$$

例えば，

$$\{\mathtt{abab}, \mathtt{ab}\} \ominus \{\mathtt{aa}, \mathtt{b}\} = (\mathtt{abab} \odot \mathtt{aa}) \cup (\mathtt{abab} \odot \mathtt{b}) \cup (\mathtt{ab} \odot \mathtt{aa}) \cup (\mathtt{ab} \odot \mathtt{b}) = \{\mathtt{bb}, \mathtt{aab}, \mathtt{aba}, \mathtt{a}\}$$

である．

以下の問いに答えよ．

(1) $\{\mathtt{ababa}, \mathtt{abb}\} \ominus \{\mathtt{ab}, \mathtt{bb}\}$ を求めよ．

(2) L_0 が正規表現 $(\mathtt{ab})^*$ で表される言語であるとき，$L_0 \ominus \{\mathtt{aa}\}$ を正規表現で表せ．

(3) L_1, L_2 がそれぞれ決定性有限オートマトン A_1, A_2 によって受理される，Σ 上の言語であるとする．$L_1 \ominus L_2$ を受理する非決定性有限オートマトンを A_1, A_2 から構成する方法を示せ．ただし ϵ 遷移を用いてもよいものとする．

(4) 以下の命題が真であるか否かを答え，真であればその証明の概略（$L_1 \ominus L_2$ を生成する文脈自由文法の構成法を簡単な説明とともに示せばよい）を，そうでなければ簡単な説明とともに反例を示せ．

> **命題.** Σ 上のすべての言語 $L_1, L_2 \subseteq \Sigma^*$ について，L_1 が正規言語かつ L_2 が文脈自由言語ならば，$L_1 \ominus L_2$ は文脈自由言語である．

Problem 1

Let Σ be $\{\mathtt{a}, \mathtt{b}\}$. For words $w_1, w_2 \in \Sigma^*$ over Σ, we define $w_1 \odot w_2 \subseteq \Sigma^*$ by:

$$w_1 \odot w_2 = \{v_1 \cdots v_{k+1} \mid \\ k \geq 0 \wedge a_1, \ldots, a_k \in \Sigma \wedge v_1, \ldots, v_{k+1} \in \Sigma^* \wedge w_1 = v_1 a_1 v_2 \cdots v_k a_k v_{k+1} \wedge w_2 = a_1 \cdots a_k\}.$$

In other words, $w_1 \odot w_2$ denotes the set of words that can be obtained from w_1 by removing the subsequence w_2. For example, $\mathtt{abababa} \odot \mathtt{bb} = \{\mathtt{aaaba}, \mathtt{aabaa}, \mathtt{abaaa}\}$.

For languages $L_1, L_2 \subseteq \Sigma^*$ over Σ, we define $L_1 \ominus L_2 \subseteq \Sigma^*$ by:

$$L_1 \ominus L_2 = \bigcup_{w_1 \in L_1, w_2 \in L_2} w_1 \odot w_2.$$

For example,

$$\{\mathtt{abab}, \mathtt{ab}\} \ominus \{\mathtt{aa}, \mathtt{b}\} = (\mathtt{abab} \odot \mathtt{aa}) \cup (\mathtt{abab} \odot \mathtt{b}) \cup (\mathtt{ab} \odot \mathtt{aa}) \cup (\mathtt{ab} \odot \mathtt{b}) = \{\mathtt{bb}, \mathtt{aab}, \mathtt{aba}, \mathtt{a}\}.$$

Answer the following questions.

(1) Compute $\{\mathtt{ababa}, \mathtt{abb}\} \ominus \{\mathtt{ab}, \mathtt{bb}\}$.

(2) Let L_0 be the language denoted by the regular expression $(\mathtt{ab})^*$. Express $L_0 \ominus \{\mathtt{aa}\}$ as a regular expression.

(3) Let L_1 and L_2 be the languages over Σ accepted by deterministic finite automata A_1 and A_2 respectively. Show how to construct a finite non-deterministic automaton that accepts $L_1 \ominus L_2$ by using A_1 and A_2. Here, you may use ϵ-transitions.

(4) Answer whether the following proposition is true or not. If it is true, then give a proof outline (it is sufficient to show how to construct a context-free grammar that generates $L_1 \ominus L_2$ with a brief explanation); otherwise, give a counterexample with a brief explanation.

> **Proposition.** For all languages L_1 and L_2 over Σ, if L_1 is a regular language and L_2 is a context-free language, then $L_1 \ominus L_2$ is a context-free language.

問題 2

オペレーティングシステムのページングに関する以下の問いに答えよ．ただし，仮想アドレスはバイトアドレッシングを用い，ページテーブルは単一レベル構成とする．すべてのページフレームおよび TLB エントリは初期状態で空であるものとする．明示的に定義されていない処理にかかる時間は無視してよい．

(1) 以下の各構成において，仮想アドレス空間の総ページ数，ページテーブル全体のサイズ，1 ページに格納できるページテーブルエントリ (PTE) の数を求めよ．ただし，$v \geq p \geq s \geq 1$ とする．

  (a) 仮想アドレス空間: 16 ビット，ページサイズ: 256 バイト，PTE サイズ: 2 バイト

  (b) 仮想アドレス空間: 24 ビット，ページサイズ: 4096 バイト，PTE サイズ: 4 バイト

  (c) 仮想アドレス空間: v ビット，ページサイズ: 2^p バイト，PTE サイズ: 2^s バイト

(2) 以下のページ参照列に対して，ページフレーム数が 3 および 4 の場合のそれぞれについて，ページフォールトの合計回数を求めよ．ただし，ページ置換アルゴリズムは First-In, First-Out (FIFO) とする．

  2, 4, 7, 5, 2, 4, 8, 2, 4, 7, 5, 8, 1, 6, 3

(3) ページフレーム数が 8，ページ置換アルゴリズムが FIFO の場合に，Translation Look-aside Buffer (TLB) を追加した構成を考える．ただし，TLB は 4 エントリのフルアソシアティブ構成とし，置換アルゴリズムは Least Recently Used (LRU) とする．アクセス時間は，TLB が 10 ns，メモリが 100 ns とし，ページフォールト処理時間は 1 μs とする．ページ参照列が問い (2) と同じであるとき，以下の問いに答えよ．

  (a) TLB のヒット数を求めよ．

  (b) TLB のヒット／ミスおよびページフォールトの有無によって起こり得るそれぞれの組み合わせについて，1 回のメモリアクセスにかかる時間を求めよ．ただし，ページフォールト処理は，ページのロード，ページテーブルの更新，および対応する TLB エントリのフラッシュのみを行い，その後は TLB 参照からアクセスを再開する．

  (c) 実効メモリアクセス時間を有効数字 3 桁で求めよ．

(4) 問い (1)(a) の構成において，ページフレーム数が 4，ページ置換アルゴリズムが FIFO の場合に，問い (3) で述べた TLB を追加した構成を考える．TLB アクセス時間，メモリアクセス時間，およびページフォールト処理時間は問い (3) と同じであると仮定する．このとき，以下の仮想アドレス（16 進数表記）に順番にアクセスした場合の，実効メモリアクセス時間を有効数字 3 桁で求めよ．

  0x000, 0x100, 0x200, 0x300, 0x050, 0x150, 0x400, 0x200

Problem 2

Answer the following questions on paging in operating systems. Assume that virtual addresses use byte addressing, and that the page table has a single-level structure. Assume that all page frames and TLB entries are initially empty. You may ignore the time required for any operations not explicitly defined.

(1) For each of the following configurations, calculate the total number of pages in the virtual address space, the total size of the page table, and the number of page table entries (PTEs) that can fit in one page. Assume that $v \geq p \geq s \geq 1$.

   (a) Virtual address space: 16 bits, page size: 256 bytes, PTE size: 2 bytes

   (b) Virtual address space: 24 bits, page size: 4096 bytes, PTE size: 4 bytes

   (c) Virtual address space: v bits, page size: 2^p bytes, PTE size: 2^s bytes

(2) For the following page reference string, calculate the total number of page faults for each of the cases where the number of page frames is 3 and 4. Assume that the page replacement algorithm is First-In, First-Out (FIFO).

   2, 4, 7, 5, 2, 4, 8, 2, 4, 7, 5, 8, 1, 6, 3

(3) Consider a configuration where the number of page frames is 8, and the page replacement algorithm is FIFO, with the addition of a Translation Look-aside Buffer (TLB). Assume that the TLB is fully associative with 4 entries, and that it uses the Least Recently Used (LRU) replacement algorithm. The access times are 10 ns for the TLB and 100 ns for memory. The page fault handling time is 1μs. When the page reference string is the same as in Question (2), answer the following questions.

   (a) Calculate the number of TLB hits.

   (b) For each of the possible combinations of TLB hit/miss and the presence or absence of a page fault, calculate the time required for a single memory access. Assume that page fault handling involves only loading the page, updating the page table, and flushing the corresponding TLB entry, after which the access restarts from the TLB lookup.

   (c) Calculate the effective memory access time to three significant digits.

(4) Consider a configuration based on Question (1)(a), where the number of page frames is 4 and the page replacement algorithm is FIFO, with the addition of the TLB described in Question (3). Assume that the TLB access time, memory access time, and page fault handling time are the same as in Question (3). In this case, calculate the effective memory access time to three significant digits when accessing the following virtual addresses (in hexadecimal) in order.

   0x000, 0x100, 0x200, 0x300, 0x050, 0x150, 0x400, 0x200

問題 3

毎クロックサイクル最大 1 命令を発行するパイプラインプロセッサ P を考える．P は，命令フェッチ (IF) ステージ，命令デコード (ID) ステージ，実行 (EX) ステージ，メモリアクセス (MA) ステージ，ライトバック (WB) ステージの 5 つのパイプラインステージを備える．P は x0, x1, x2, x3 の 4 個のレジスタで構成されるレジスタファイルを持ち，ID ステージでレジスタファイルを読み出す．P は，EX ステージと MA ステージの間のパイプラインレジスタから，および，MA ステージと WB ステージの間のパイプラインレジスタから，EX ステージにデータを渡すフォワーディング機構を備える．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，被ロードデータハザードにより，1 クロックサイクルの間ストールする．それ以外の理由では P はストールしない．

以下の問いに答えよ．

(1) 最大クロック動作周波数で動作する P 上で，あるプログラム S を実行すると，実行命令数が N であった．すべての実行命令のうち，ロード命令の割合が R_L であり，すべてのロード命令のうち，ロード命令の直後の命令がロード結果を利用する割合は R_U であった．P の各パイプラインステージの遅延は以下に示す通りであり，フォワーディング機構などのパイプラインステージをまたぐ回路の遅延は無視できるものとする．このときの実行時間 T を，N, R_L, R_U を用いて表せ．

- IF ステージ: 360 ピコ秒
- ID ステージ: 320 ピコ秒
- EX ステージ: 400 ピコ秒
- MA ステージ: 360 ピコ秒
- WB ステージ: 160 ピコ秒

(2) P のレジスタファイルは，1 行が 4 バイト，行数が 4 の SRAM で構成される．この SRAM のアドレスデコーダーは，2 ビットの行アドレス入力からワンホット信号を出力するイネーブル付き 2 ビットデコーダー回路である．この回路は，イネーブル入力が 1 のときのみワンホット信号を出力し，それ以外のときはすべての出力は 0 となる．NOT ゲート，2 入力 NAND ゲート，2 入力 NOR ゲートのみを用いて，このアドレスデコーダー回路を設計せよ．ただし，NOT ゲートの個数は 4 個以下，2 入力 NAND ゲートと 2 入力 NOR ゲートの個数は合計 8 個以下とすること．

(3) 2 つの 2 ビット入力値が一致する場合には 1，そうでなければ 0 を出力する比較回路をフォワーディング機構で用いる．NOT ゲート，2 入力 NAND ゲート，2 入力 NOR ゲートのみを用いて，この比較回路を設計せよ．ただし，NOT ゲートの個数は 4 個以下，2 入力 NAND ゲートと 2 入力 NOR ゲートの個数は合計 7 個以下とすること．

(4) P はフォワーディング機構を備えるにもかかわらず，被ロードデータハザードが発生してしまう理由を説明せよ．また，もし P がフォワーディング機構を持たない場合，どのような問題が発生するか，問題を引き起こす命令列の例と共に説明せよ．例で用いる各命令の振る舞いを述べること．

Problem 3

Consider a pipeline processor `P` that issues up to one instruction per clock cycle. `P` consists of the following five pipeline stages: *instruction fetch* (`IF`) stage, *instruction decode* (`ID`) stage, *execute* (`EX`) stage, *memory access* (`MA`) stage, and *write back* (`WB`) stage. `P` has a register file of four registers `x0`, `x1`, `x2`, and `x3` and reads the register file in the `ID` stage. `P` has a forwarding unit that delivers data from a pipeline register between `EX` and `MA` stages and a pipeline register between `MA` and `WB` stages to the `EX` stage. `P` stalls for one clock cycle when the next instruction of a load instruction uses the load result, due to the load-use data hazard. `P` will not stall for any other reasons.

Answer the following questions.

(1) Suppose that the number of executed instructions was N for the execution of some program `S` on `P` running at the maximum clock frequency. Suppose also that, among all the executed instructions, the proportion of load instructions was R_L, and among all the load instructions, the proportion of those whose load result was used by the next instruction was R_U. The delay of each pipeline stage of `P` is as given below, and the delay of circuits across pipeline stages, such as the forwarding unit, is negligible. Express the execution time T in terms of N, R_L, and R_U.

- `IF` stage: 360 picoseconds
- `ID` stage: 320 picoseconds
- `EX` stage: 400 picoseconds
- `MA` stage: 360 picoseconds
- `WB` stage: 160 picoseconds

(2) The register file of `P` is implemented using an SRAM with 4 rows, each containing 4 bytes. The address decoder of the SRAM is a 2-bit decoder circuit with an enable input, which outputs a one-hot signal according to the 2-bit row-address input. Note that the circuit outputs a one-hot signal when the enable input is 1; otherwise all the outputs are 0. Design the address decoder circuit. You may use only NOT gates, 2-input NAND gates, and 2-input NOR gates. The number of NOT gates should be at most four, and the total number of 2-input NAND gates and 2-input NOR gates should be at most eight.

(3) In the forwarding unit, we use a comparator circuit that outputs 1 if two 2-bit input values are the same and outputs 0 otherwise. Design the comparator circuit. You may use only NOT gates, 2-input NAND gates, and 2-input NOR gates. The number of NOT gates should be at most four, and the total number of 2-input NAND gates and 2-input NOR gates should be at most seven.

(4) Explain the reason why the load-use data hazard occurs although `P` has the forwarding unit. Explain also what problem is caused if `P` has no forwarding unit, using an example of an instruction sequence that causes the problem. Describe the behavior of each instruction used in the example.

問題 4

n 個の整数からなる配列 $A = [a_1, a_2, \ldots, a_n]$ を考える．A 中の各要素には定数時間でアクセスできるものとする．二整数の加算，減算，乗算，除算および比較は定数時間で可能であるものとする．それらの演算の際のオーバーフローは考えなくてよい．

このとき，$D_i^{(k)}$ $(0 \leq k \leq n-1,\ 1 \leq i \leq n-k)$ を以下のように k に関する帰納法によって定義する．

$$D_i^{(0)} = a_i \qquad (1 \leq i \leq n),$$

$$D_i^{(k)} = D_{i+1}^{(k-1)} - D_i^{(k-1)} \quad (1 \leq k \leq n-1,\ 1 \leq i \leq n-k).$$

以下の問いに答えよ．

(1) $n > 5$ として，$1 \leq i \leq n-5$ である任意の i に対し $D_i^{(5)}$ を $a_i,\ a_{i+1}, \ldots,\ a_{i+5}$ を用いて表した場合の a_{i+3} の係数を求めよ．

(2) $n > 1$ として，すべての整数 i $(1 \leq i \leq n-1)$ に対し

$$D_i^{(1)} < 0$$

が成り立っているとする．さらに $a_1 > 0$ かつ $a_n < 0$ が成り立っているとする．このとき，$a_i \geq 0 > a_{i+1}$ となる整数 i $(1 \leq i \leq n-1)$ を見つける $O(\log n)$ 時間のアルゴリズムを示せ．疑似コードを用いてよい．

(3) $n > 2$ として，すべての整数 i $(1 \leq i \leq n-2)$ に対し

$$D_i^{(2)} < 0$$

が成り立っているとする．このとき，配列 A 中の最大要素（$\max_{1 \leq i \leq n} a_i$ のこと）を $O(\log n)$ 時間で計算できることを示せ．なお，問い (2) の仮定は必ずしも成り立っていないものとする．

(4) n によらない（すなわち $c = O(1)$ である）整数の定数 c $(c \geq 3)$ を考える．$n > c$ として，すべての整数 i $(1 \leq i \leq n-c)$ に対し

$$D_i^{(c)} < 0$$

が成り立っているとする．配列 A 中の最大要素を $O(\log n)$ 時間で計算できることを示せ．なお，問い (2) および問い (3) の仮定は必ずしも成り立っていないものとする．

Problem 4

Consider an array $A = [a_1, a_2, \ldots, a_n]$ consisting of n integers. Assume that each element in A can be accessed in constant time. Assume that addition, subtraction, multiplication, division, and comparison of two integers can be done in constant time. You need not consider overflows of these operations.

We define $D_i^{(k)}$ $(0 \leq k \leq n-1,\ 1 \leq i \leq n-k)$ by induction on k, as follows.

$$D_i^{(0)} = a_i \qquad (1 \leq i \leq n),$$

$$D_i^{(k)} = D_{i+1}^{(k-1)} - D_i^{(k-1)} \quad (1 \leq k \leq n-1,\ 1 \leq i \leq n-k).$$

Answer the following questions.

(1) Assume that $n > 5$. Show the coefficient of a_{i+3} when $D_i^{(5)}$ is expressed with $a_i,\ a_{i+1}, \ldots,\ a_{i+5}$ for any i such that $1 \leq i \leq n-5$.

(2) Assume that $n > 1$. Assume also that

$$D_i^{(1)} < 0$$

holds for all the integers i $(1 \leq i \leq n-1)$. Assume also that $a_1 > 0$ and $a_n < 0$. Show an $O(\log n)$-time algorithm that can find the integer i $(1 \leq i \leq n-1)$ such that $a_i \geq 0 > a_{i+1}$ holds. You can use a pseudocode.

(3) Assume that $n > 2$. Assume also that

$$D_i^{(2)} < 0$$

holds for all the integers i $(1 \leq i \leq n-2)$. Show that the maximum element in the array A (i.e., $\max_{1 \leq i \leq n} a_i$) can be computed in $O(\log n)$ time. Note that the assumptions in question (2) may not hold in this question.

(4) Consider an integer constant c $(c \geq 3)$ that is independent of n (i.e., $c = O(1)$). Assume that $n > c$. Assume also that

$$D_i^{(c)} < 0$$

holds for all the integers i $(1 \leq i \leq n-c)$. Show that the maximum element in the array A can be computed in $O(\log n)$ time. Note that the assumptions in questions (2) and (3) may not hold in this question.

余白 (blank page)

計算などに使ってもよいが，切り離さないこと． Usable for memos; do not detach.

余白 (blank page)

計算などに使ってもよいが，切り離さないこと． Usable for memos; do not detach.