JOI 2020/2021 二次予選 5
スパイ 2 (Spy 2)

解説: 木ノ下 恭範

問題概要

- スパイとスパイでない人がたくさんいる
- 何人かはスパイかどうかが分かっている
- いくつかの証言がある
  - A「B はスパイである　かつ　C はスパイでない」
- スパイは必ず嘘を言う，スパイでなくても嘘を言う事もある
- それぞれの人がスパイかどうか判定してください

小課題 1

- N <= 16，M <= 100
- N がとても小さい，M も比較的小さい
- スパイかそうでないかの全てのパターンを試しても大丈夫そう
- あるパターンが情報に合致するのはどういうときか？

小課題 1

- N <= 16，M <= 100
- N がとても小さい，M も比較的小さい
- スパイかそうでないかの全てのパターンを試しても大丈夫そう
- あるパターンが情報に合致するのはどういうときか？
- 証言 (A, B, C) に合致するパターンを考える
- A がスパイでない場合，証言の真偽は何でも良いので OK
- A がスパイの場合，B がスパイでないか C がスパイなら OK
- O((N + M) 2^N) → 7 点

小課題 2

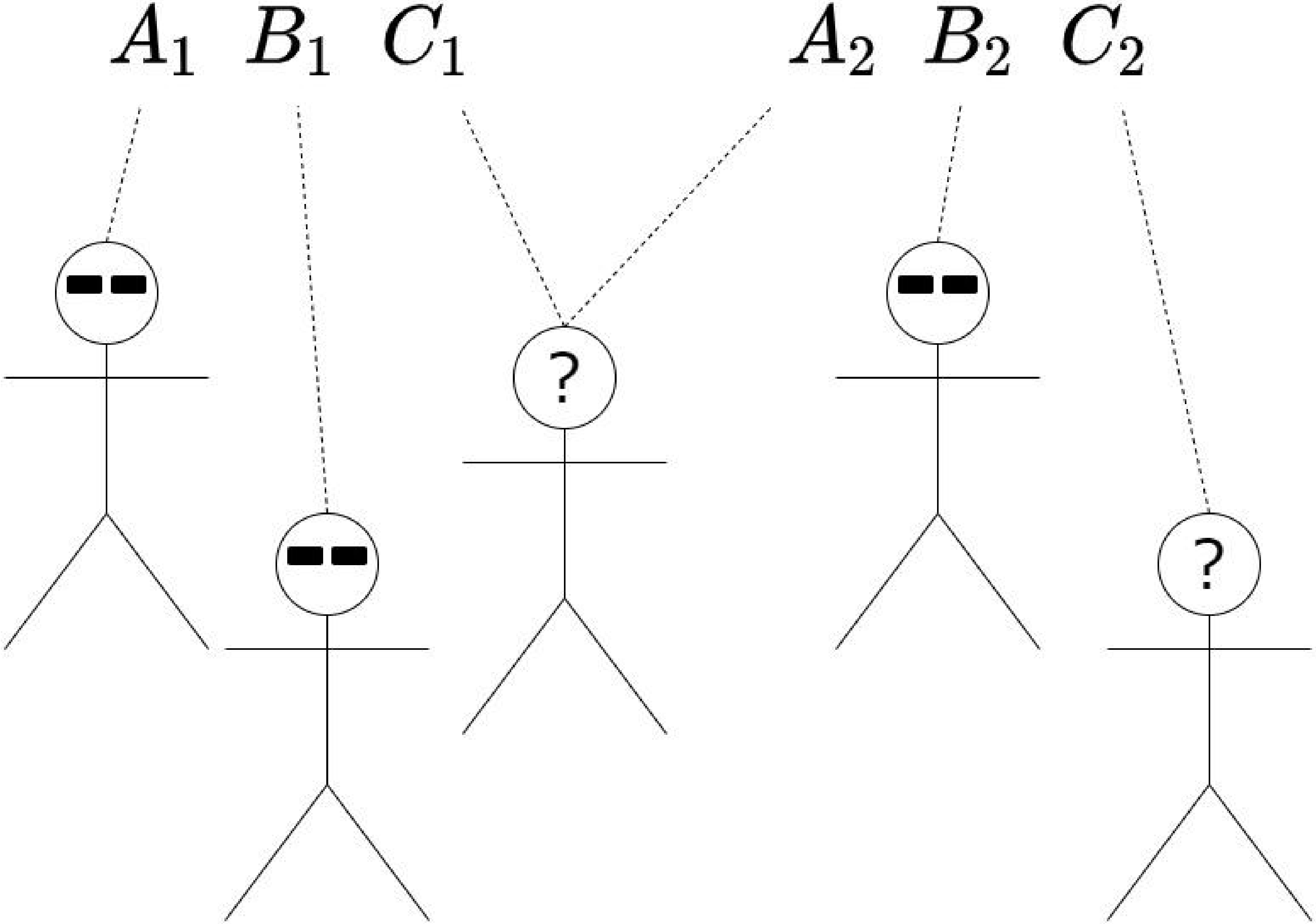
- N <= 3000，M <= 3000
- N も M も大きいので，問題の性質をよく観察することが必要
- 証言 (A, B, C) と合致しないのは
  - A がスパイ
  - B がスパイ
  - C がスパイでない

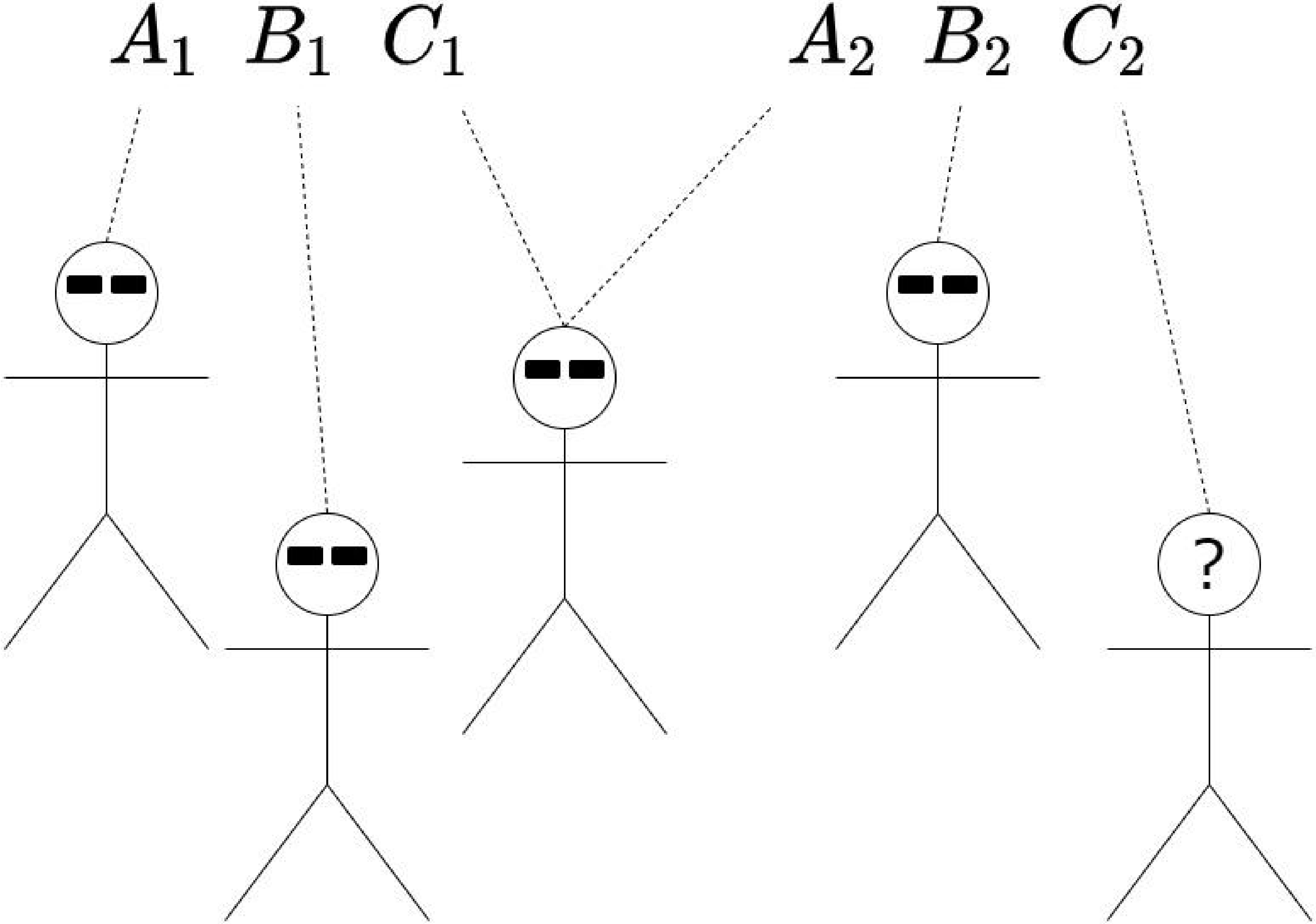
  の全てが満たされるとき，かつそのときに限る

小課題 2

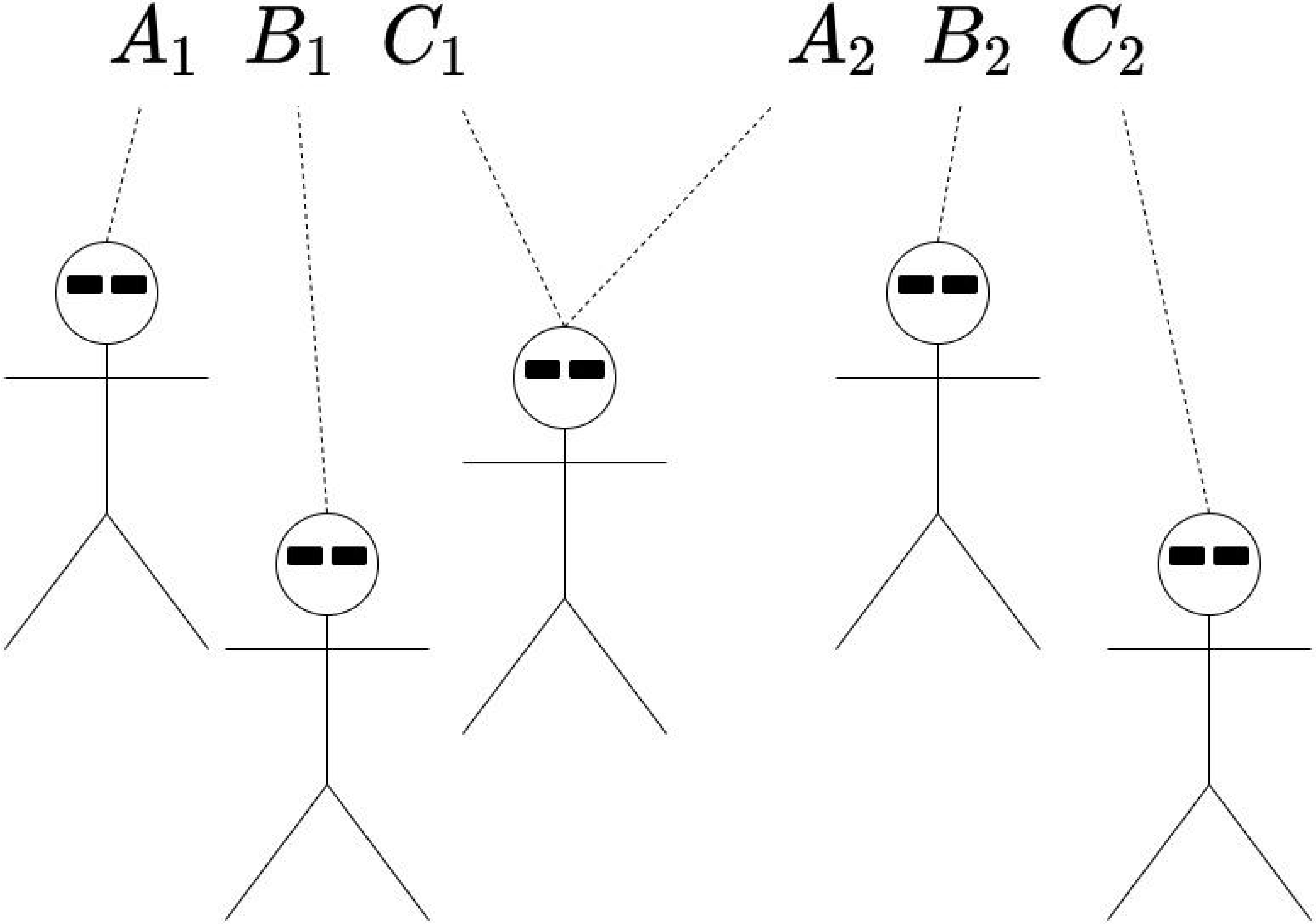
- N <= 3000， M <= 3000
- N も M も大きいので，問題の性質をよく観察することが必要
- 証言 (A, B, C) と合致しないのは
  - A がスパイ
  - B がスパイ
  - C がスパイでない

  の全てが満たされるとき，かつそのときに限る
- A がスパイで B もスパイなら，C は必ずスパイになる
- 確定した人は次々にスパイにしてしまえばよい


A_1 B_1 C_1
A_2 B_2 C_2
?
?



A_1 B_1 C_1
A_2 B_2 C_2
?


A_1 B_1 C_1
A_2 B_2 C_2

小課題 2

- もう確定する人がいないとき，どうすればよいか？
- 実は，残りの未確定の人は全てスパイでないとしてしまってよい

- これによって証言 (A, B, C) と合致しなくなるのはどんなときか？
  - A がスパイと確定していて
  - B もスパイと確定していて
  - C が未確定
- これは C がスパイ確定のパターンなので，既に処理されている

小課題 2

- 誰も新たに確定しなくなるまで，全ての証言をチェックすることを繰り返す
- 未確定の人が残ったら，スパイでないとする
- 最後に矛盾してないかチェック
- 1 周チェックするのが O(M)
- 最後の 1 周を除いて，少なくとも 1 人の情報が新しく確定する
  → 高々 N + 1 周で終わる
- O(NM) → 7 + 38 = 45 点

小課題 3

- N <= 300000， M <= 300000
- 更に高速化しなければいけない
- 証言 (A, B, C) によって C がスパイ確定する可能性があるのは，A か B が新たにスパイと確定した後 (+ 最初から)
- 人 X がスパイと確定したとき，A = X または B = X である証言だけを再チェックする

小課題 3

- N <= 300000，M <= 300000
- 更に高速化しなければいけない
- 証言 (A, B, C) によって C がスパイ確定する可能性があるのは，A か B が新たにスパイと確定した後 (+ 最初から)
- 人 X がスパイと確定したとき，A = X または B = X である証言だけを再チェックする
- 計算量は？
- 1 つの証言は高々 2 回しか再チェックされない
- 適切に実装すると O(N + M) → 7 + 38 + 55 = 100 点

小課題 3

- 適切な実装とは？
- 幅優先探索や深さ優先探索のイメージで，「スパイ確定処理を待つ人々」をキューやスタックで管理する
- X のスパイ確定処理: (X, B, C) や (A, X, C) のような証言全てについて条件をチェックし，C がスパイと確定するなら C をキューに追加する
- それぞれの人についてリストを用意して，各証言 (A, B, C) について，その情報を A のリストと B のリストに追加しておくとよい（隣接リストのイメージ）

補足

- 変数に true と false を割り当てて条件を満たすようにする問題を SAT (satisfiability problem) と呼ぶ
- 今回の問題は (¬A ∨ ¬B ∨ C) ∧ .... の形の条件
- どの括弧の中も高々 1 つの変数を除いて否定 (¬) されているものは Horn SAT と呼ばれ，この問題と概ね同じアルゴリズムを用いて高速に解くことが出来る
- 一般の SAT を高速に解くことはとても難しい