Time Bounded Complexity Classes, Time Hierarchy Theorem

Factoring

- **Instance**: Positive integers N, M (in binary)
- **Problem**: Does N have a prime factor in the range $[1..M]$?

Show that Factoring is in NP. Useful facts:

- Any integer > 1 has a unique factorization as the product of primes.
- Primality testing is in P [Agrawal-Kayal-Saxena 2004].

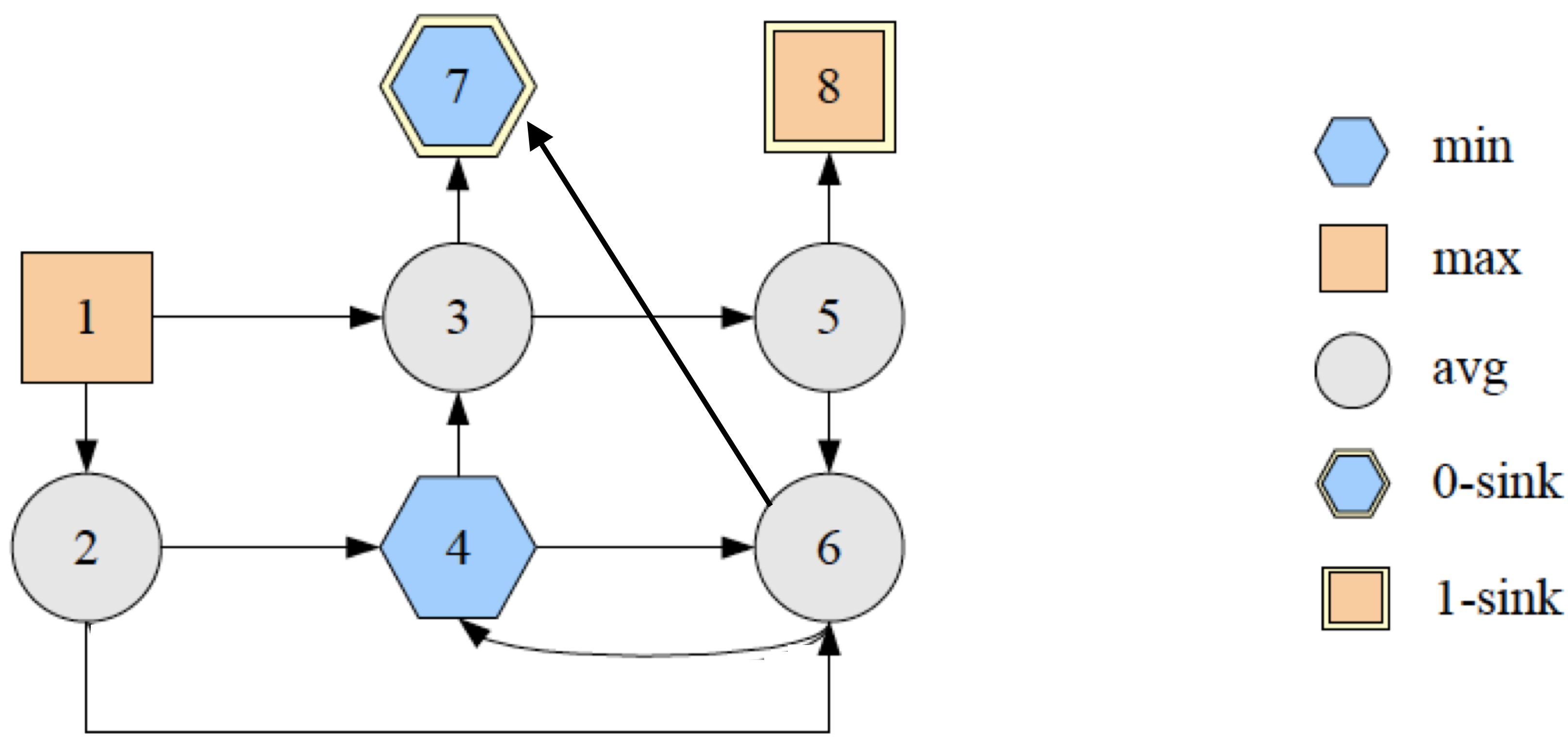
Factoring

- **Instance**: Positive integers N, M (in binary)
- **Problem**: Does N have a prime factor in the range $[1..M]$?

Similar reasoning shows that Factoring is in co-NP!

Simple Stochastic Games (SSGs)

- A SSG is a directed graph with a start node, two sink nodes, and a partition of V into three sets: Min, Max, and Average.

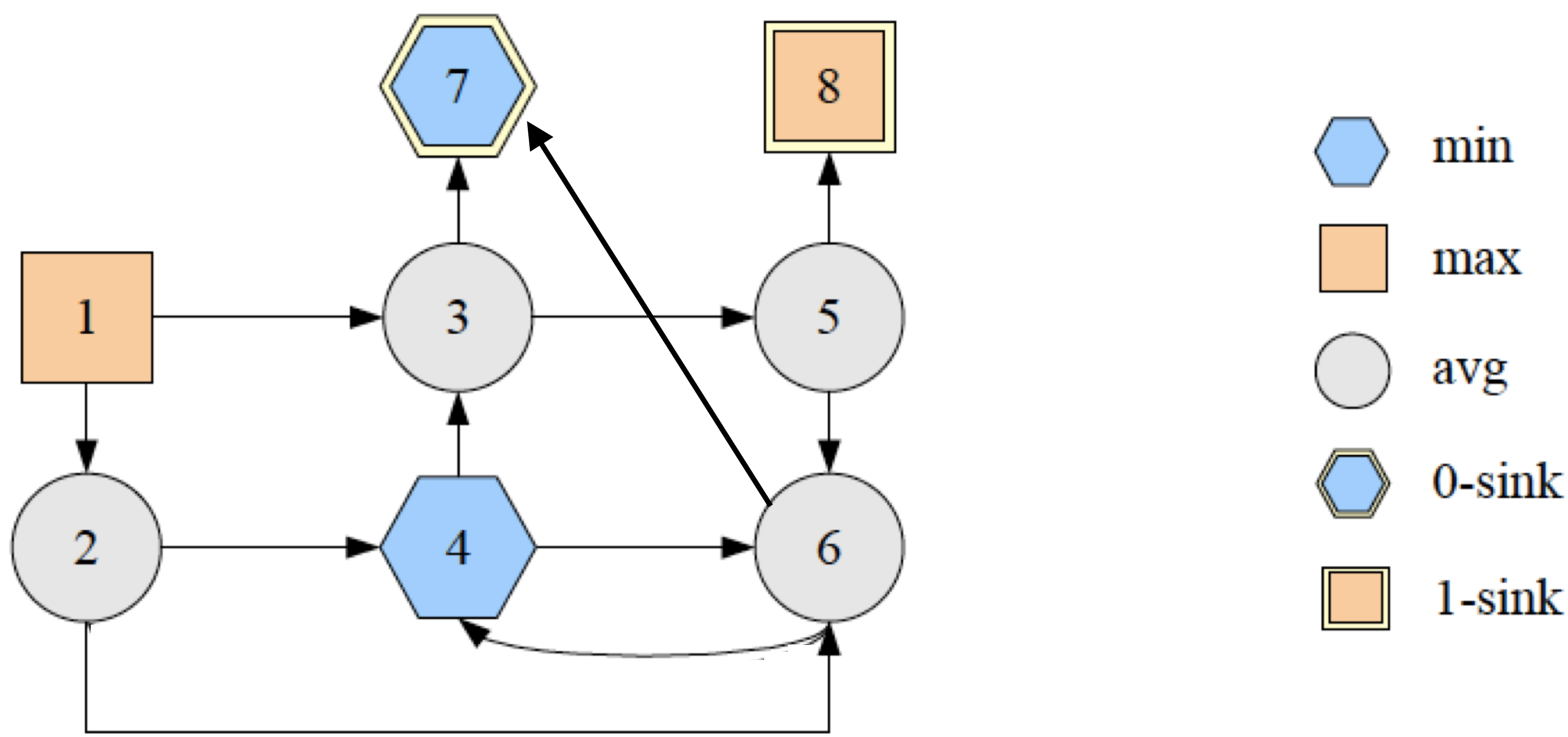


Simple Stochastic Games (SSGs)

- A SSG is a directed graph with a start node, two sink nodes, and a partition of V into three sets: Min, Max, and Average.
- The game starts with a token at the start node.
- From a node v, the token moves to a successor u of v chosen as follows:
  - If $v \in$ Max then player 1 chooses u.
  - If $v \in$ Min, then player 0 chooses u.
  - If $v \in$ Average then u is chosen randomly.
- Player 0 (1) wins if the token reaches the 0-sink (1-sink)

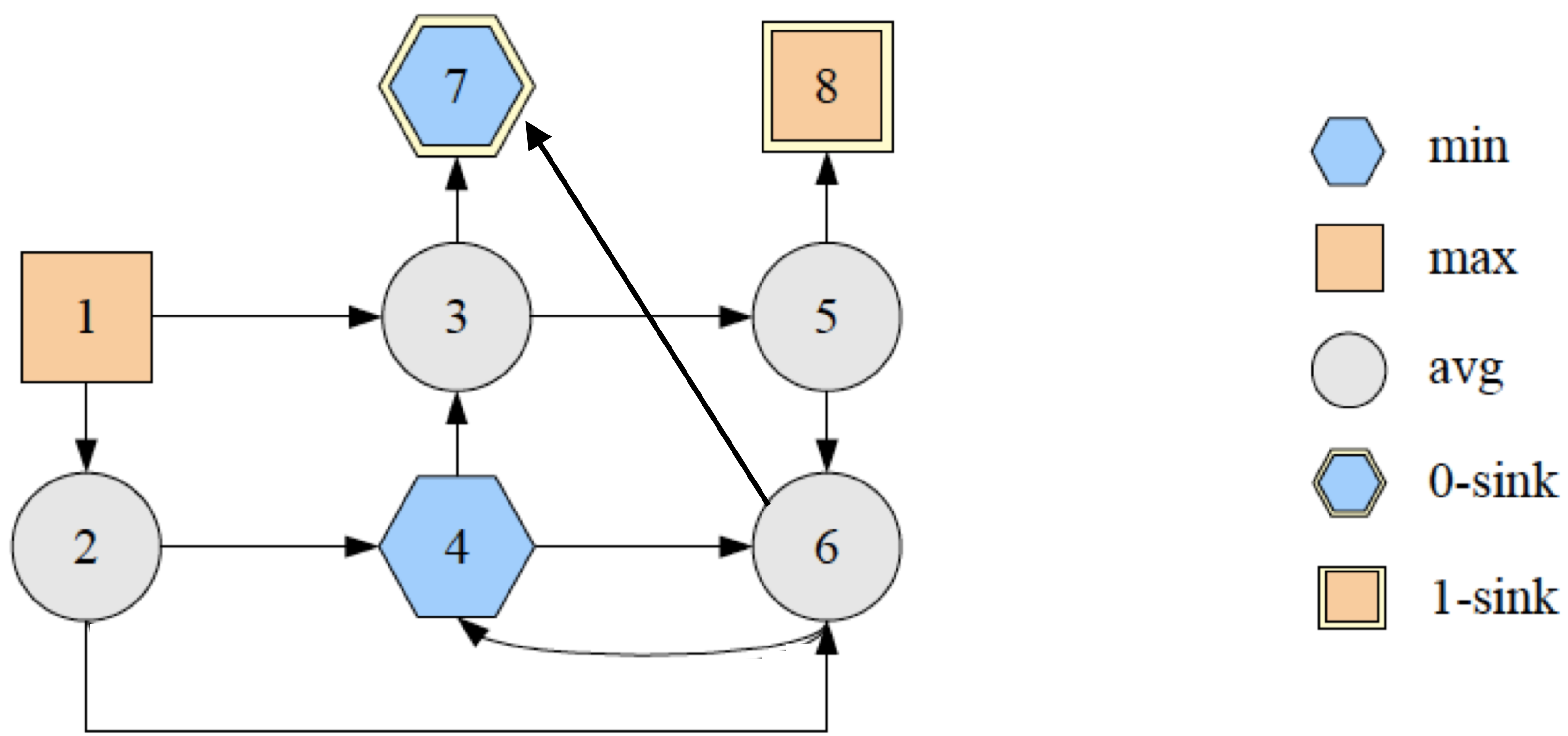
Simple Stochastic Games (SSGs)

- **Instance:** A Simple Stochastic Game
- **Problem:** Does player 1 win with probability > 1/2?



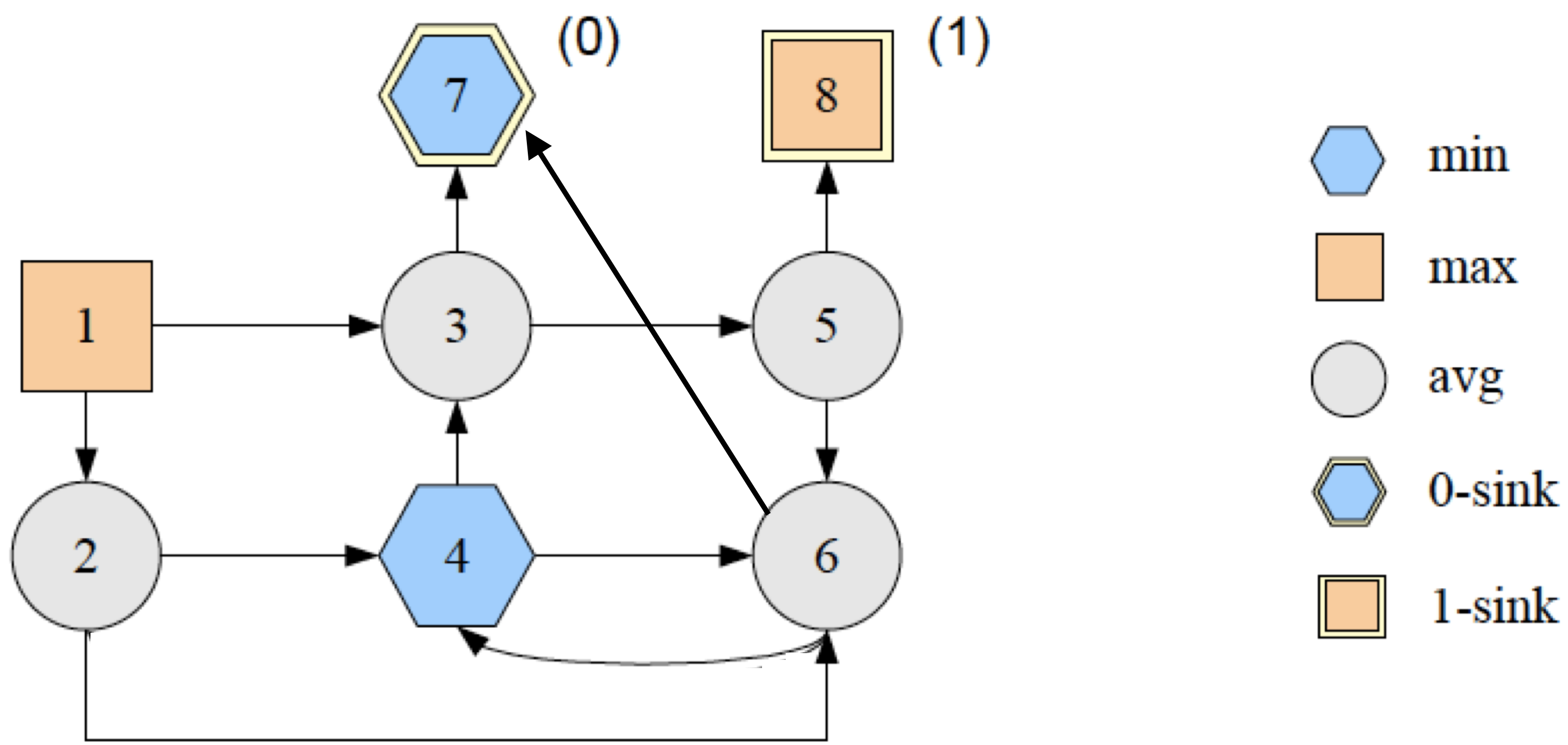
Simple Stochastic Games (SSGs)

Let's label each node with the probability that player 1 wins if the game starts at that node.



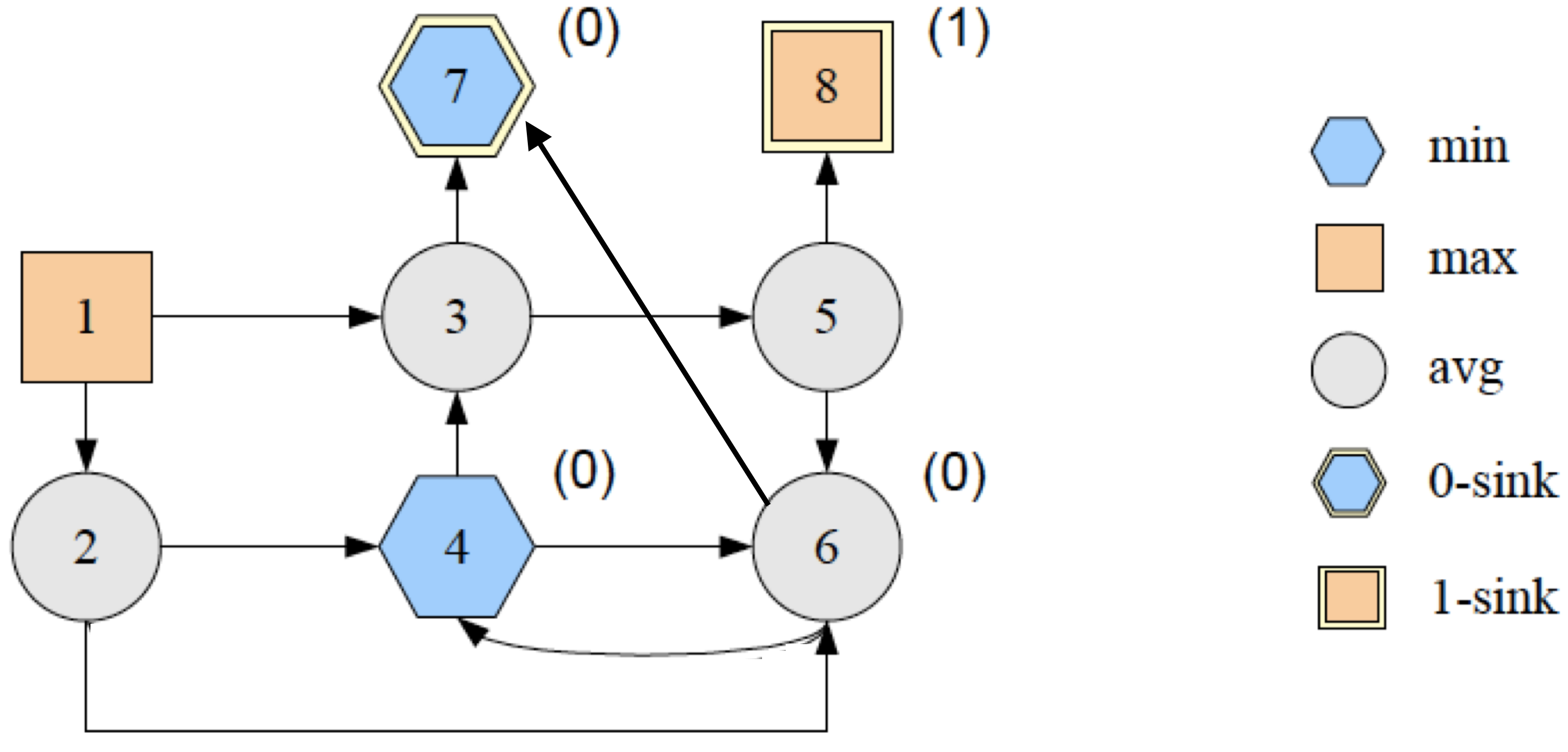
Simple Stochastic Games (SSGs)

Let's label each node with the probability that player 1 wins if the game starts at that node.



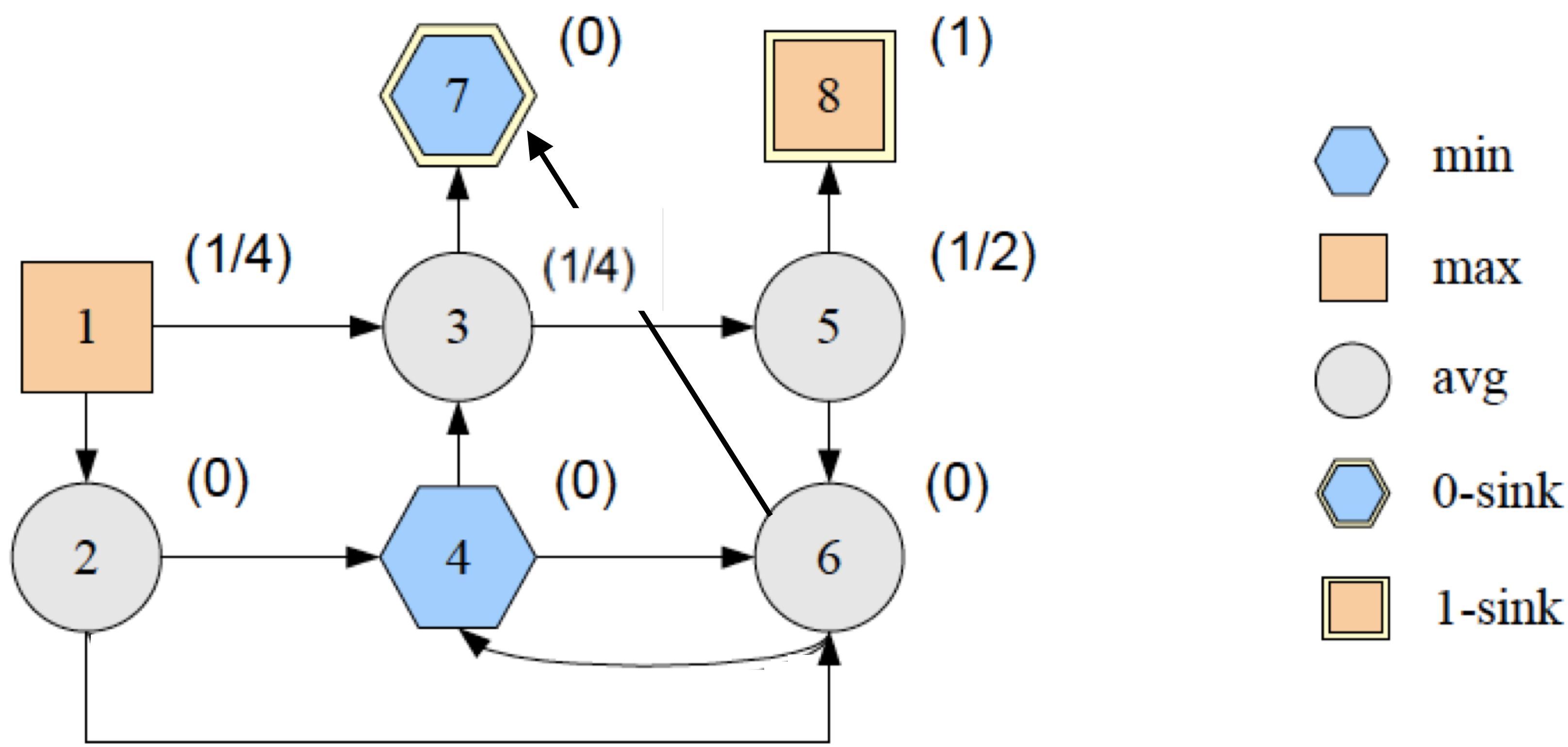
Simple Stochastic Games (SSGs)

Let's label each node with the probability that player 1 wins if the game starts at that node.



Simple Stochastic Games (SSGs)

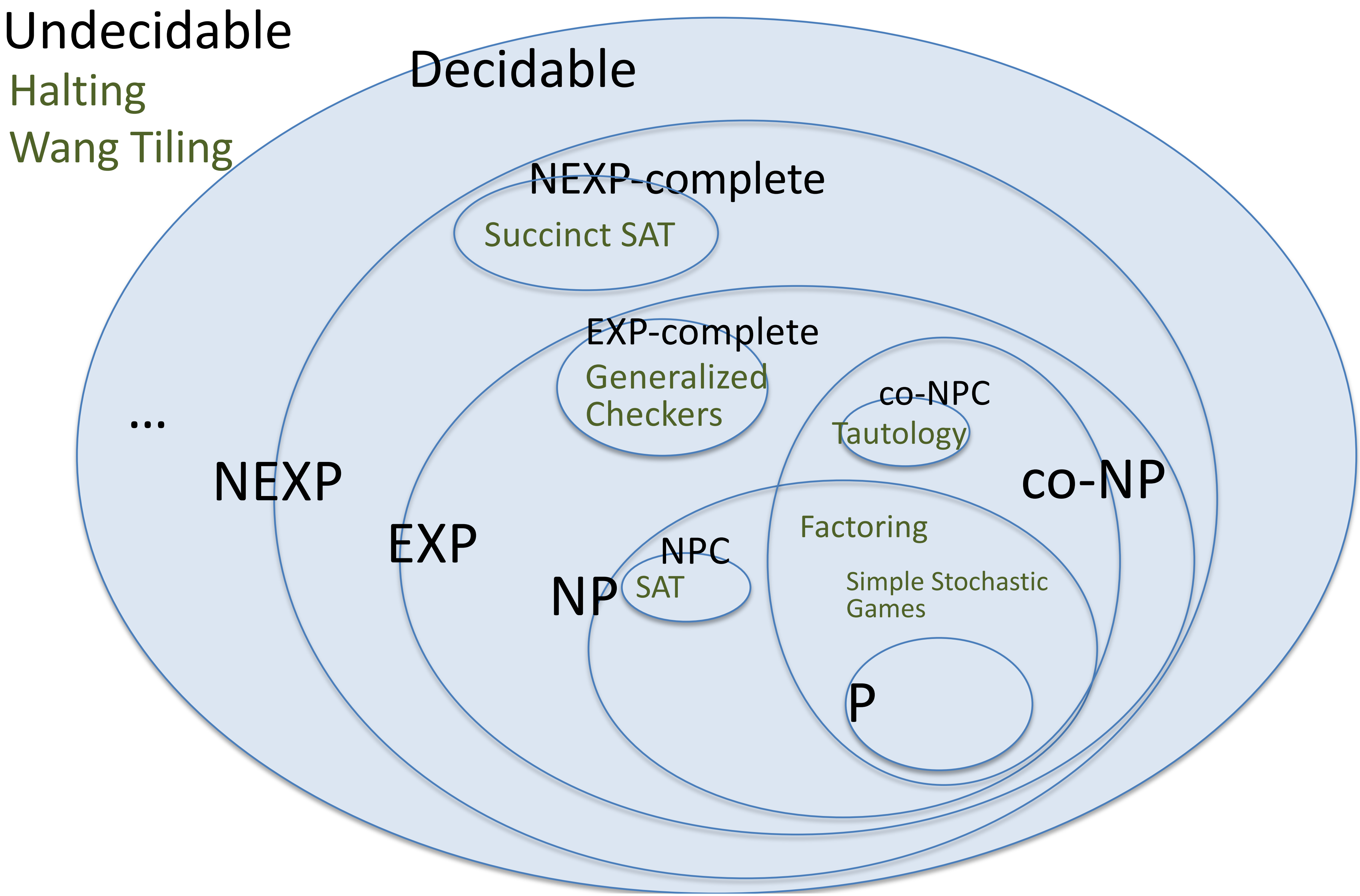
Let's label each node with the probability that player 1 wins if the game starts at that node.



Simple Stochastic Games (SSGs)

- A *certificate* assigns node labels (prob of winning):
  - label of 1-sink is 1 and value of 0-sink is 0
  - label of a max/min/average node is the max/min/average of its children's labels
- SSG's that halt with probability 1 (regardless of players' strategies) have *unique certificates*
- By guessing a certificate, it is possible to verify in polynomial time whether or not player 1 wins, so SSG is in NP $\cap$ co-NP.

Summary: New Problems and Complexity Classes



A Time Hierarchy Theorem

Theorem (rough version): If $f(n)$ "<<" $g(n)$ then

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$$

A Time Hierarchy Theorem

Theorem (rough version): If $f(n)$ "<<" $g(n)$ then

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$$

- Is Succinct SAT or Generalized Checkers in P?
- No, since these problems are hard for EXP. If they were in P then we would have EXP = P, contradicting the Time Hierarchy theorem

A Time Hierarchy Theorem

Theorem (rough version): If $f(n)$ “<<“ $g(n)$ then

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$$

A Time Hierarchy Theorem

Theorem (rough version): If $f(n)$ "<<" $g(n)$ then

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$$

Proof ideas:

- *Diagonalization*: construct M_g that runs in time $O(g(n))$, so that for every TM M_x that runs in $O(f(n))$ time, M_g *does the opposite* of M_x on some input w.

A Time Hierarchy Theorem

Theorem (rough version): If $f(n)$ "<<" $g(n)$ then

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$$

Proof ideas:

- *Diagonalization*: construct M_g that runs in time $O(g(n))$, so that for every TM M_x that runs in $O(f(n))$ time, M_g *does the opposite* of M_x on some input w.
- *Challenges:* Making sure that
  - M_g halts within $O(g(n))$ time …
  - while still having enough time to fully simulate M_x …
  - on some large enough input w

A Time Hierarchy Theorem

Theorem (rough version): If $f(n)$ "<<" $g(n)$ then

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$$

Proof ideas:

- *Diagonalization*: construct M_g that runs in time $O(g(n))$, so that for every TM M_x that runs in $O(f(n))$ time, M_g *does the opposite* of M_x on some input w.
- *Challenges:* Making sure that
  - M_g halts within $O(g(n))$ time …
  - while still having enough time to fully simulate M_x …
  - on some large enough input w
- See handout for details (covered in class)

Next Class

- Space bounded complexity classes
- See Chapter 4.1, 4.2 of Arora-Barak