Problem Solving in Computer Science - Week 1

Scribe on duty: Paul Dütting

September 19, 2008

1 Defining Problem 1

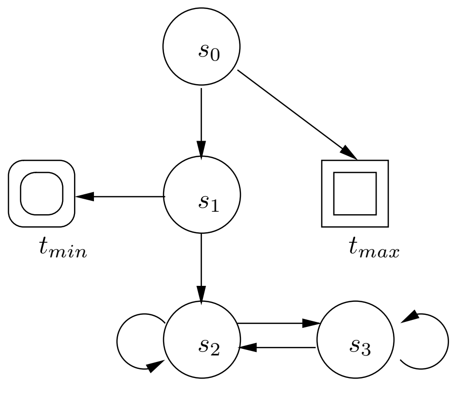
A concise definition of so-called *simple stochastic games* (SSG) and the related decision problem can be found on the home page of this course: `http://mtc.epfl.ch/courses/ProblemSolving-2008/`.

1.1 General Remarks

When writing a problem definition one can either (1) seperate individual definitions from the main text (e.g. by using dedicated environments) or (2) integrate individual definitions in the main text (e.g. by using italics). While (1) is regarded as more formal, (2) typically improves fluency.

1.2 Specific Remarks

Unlike the solution discussed in class the suggested solution uses two distinct *successor functions* to establish connections between subsequent states of a SSG game. Thus, there is no need to bound the outdegree of a state explicitly. Another "trick" used by the suggested solution is to introduce the notion of *plays*. A *winning play* is a finite sequence of states that ends up in one of the terminal states. A game can either be *stopping* or *non-stopping*. By restricting itself to stopping games the suggested solution solves the problem of defining a probability space in an elegant manner.



winning games:
$\langle s_0, s_1, t_{min} \rangle$
$\langle s_0, t_{max} \rangle$

probabilities:
$Pr(\langle s_0, s_1, t_{min} \rangle) = \frac{1}{4}$
$Pr(\langle s_0, t_{max} \rangle) = \frac{1}{2}$

conclusion:
non-stopping

Figure 1: An example of a *non-stopping game*.

2 Gimbert and Horn's Approach to Problem 1

This section presents the approach taken by Gimbert and Horn [3]. Based on permutations $p = \langle s_0, s_1, ..., s_m \rangle$ (where $m = |S_{ran}| - 1$) of the random states S_{ran}, Gimbert and Horn define so-called p-strategies σ_p. They then show that for every SSG there exists at least one permutation p for which the corresponding p-strategy is optimal. Apparently, this reduces the search space to $(m+1)!$ strategies which makes exhaustive search an eligible option.

2.1 Winning Sets

Every permutation $p = \langle s_0, s_1, \ldots, s_m \rangle$ defines so-called *winning sets* $W_-, W_0, W_1, \ldots W_{m+1}$ where $W_0, \ldots, W_m \subset S$ and $W_-, W_{m+1} \subset S^+$ are such that

- W_- is the set of all states from which the min player can force the token to reach the terminal state t_{min} or remain within W_- forever,
- W_i for $1 \leq i \leq m$ is the set of all states from which the max player can force the token to reach s_i or a state in $\cup_{j=i}^{m} W_{j+1}$,
- W_{m+1} is the set of all states from which the max player can force the game to the terminal state t_{max}.

The following figure illustrates the decomposition of S into W_-, W_0, W_1, W_2 for a SSG with $m+1 = 2$ random states s_0 and s_1 and the (randomly selected) permutation $p = \langle s_0, s_1 \rangle$.

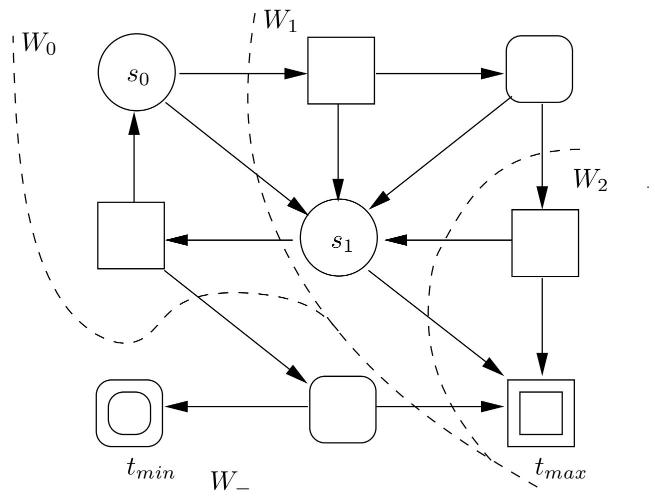


Figure 2: Winning sets.

The sets W_i can be computed by a backward traversal of the graph that iteratively adds or rejects the visited states. A max state is added if at least one of its succesors is part of W_i. A min state is added only if both of its succesors are part of W_i. The following figure depicts the computation of W_2 for the same SSG as above.

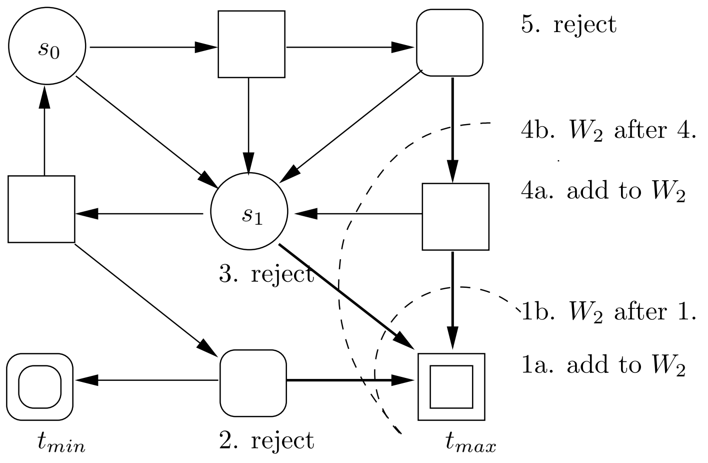


Figure 3: Computation of Winning Sets.

2.2 Derived Strategies

Given a permutation p of the random states and a decomposition of S^+ into $W_-, W_0, \ldots, W_{m+1}$ the p-strategy of the max player is as follows: Always try to attract the game to the "highest" W_i possible. Once a state in W_{m+1} is reached the game is won (no matter how the min player plays). However, as long as the current state is not in W_{m+1} it is well possible that the game is forced back to a state that belongs to a "lower" W_i as the following figure illustrates.

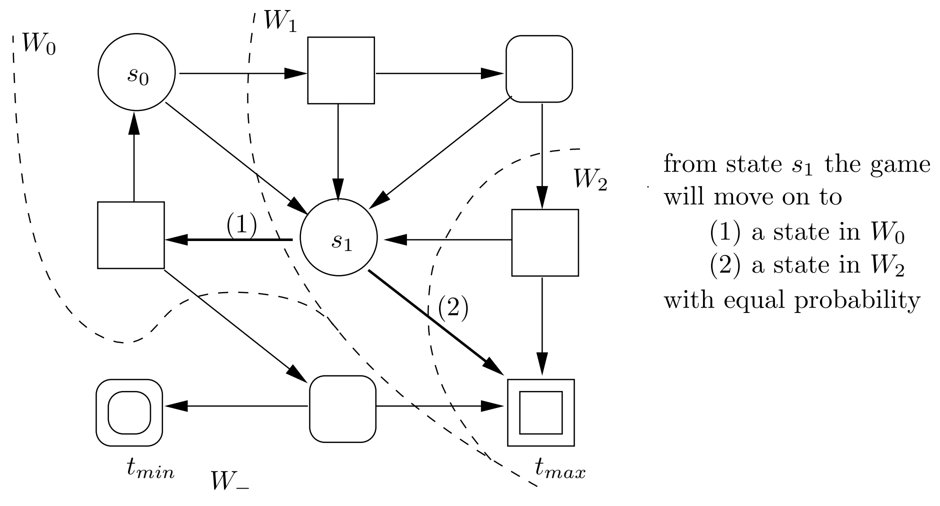


Figure 4: Interpretation of Winning Sets.

2.3 Computation of Optimal Strategies

Gimbert and Horn show that for every SSG there exists a permutation p for which the corresponding p-strategy σ_p is optimal. This motivates the following algorithm: For all possible permutations p of S_{rand} compute σ_p and test for optimality. Stop once the optimal strategy σ_p is found and output σ_p.

3 TO DO

The algorithm presented by Gimbert and Horn relies on an *exhaustive search* among the p-strategies that correspond to the $|S_{ran}|!$ possible permutations p of the random states S_{ran}. A heuristic that constructs *good permutations*, e.g. by iteratively improving a random initial permutation, will most likely lead to an improved algorithm. Your task will be to devise and evaluate such a heuristic. In short,

- understand the approach taken by Gimbert and Horn,
- figure out what makes a permutation a good permutation, and
- come up with an heuristic that finds such permutations.

References

[1] A. Condon. *The Complexity of Stochastic Games.* Journal of Information and Computation, pp. 203–224, 1992.

[2] A. Condon. *On algorithms for simple stochastic games.* Advances in Computational Complexity Theory, pp. 51–73, 1993.

[3] H. Gimbert and F. Horn. *Simple Stochastic Games with Few Random Vertices Are Easy to Solve.* FoSSaCS, pp. 5–19, 2008.