COS 445 - PSet 5

Due online Monday, April 26th at 11:59 pm.

Instructions:

- Some problems will be marked as no collaboration problems. This is to make sure you have experience solving a problem start-to-finish by yourself in preparation for the midterms/final. You cannot collaborate with other students or the Internet for these problems (you may still use the referenced sources and lecture notes). You may ask the course staff clarifying questions, but we will generally not give hints.
- Submit your solution to each problem as a **separate PDF** to codePost. Please make sure you're uploading the correct PDFs![1] If you collaborated with other students, or consulted an outside resource, submit a (very brief) collaboration statement as well. Please anonymize your submission, although there are no repercussions if you forget.
- This cheatsheet gives problem solving tips, and guidelines for a "good proof" or "partial progress": `http://www.cs.princeton.edu/~smattw/Teaching/cheatsheet445.pdf`.
- Please reference the course collaboration policy here: `http://www.cs.princeton.edu/~smattw/Teaching/infosheet445sp21.pdf`.

Instructions:

- Some problems will be marked as no collaboration problems. This is to make sure you have experience solving a problem start-to-finish by yourself in preparation for the midterms/final. You cannot collaborate with other students or the Internet for these problems (you may still use the referenced sources and lecture notes). You may ask the course staff clarifying questions, but we will generally not give hints.
- Submit your solution to each problem as a **separate PDF** to codePost. Please make sure you're uploading the correct PDFs![2] If you collaborated with other students, or consulted an outside resource, submit a (very brief) collaboration statement as well. Please anonymize your submission, although there are no repercussions if you forget.
- This cheatsheet gives problem solving tips, and guidelines for a "good proof" or "partial progress": `http://www.cs.princeton.edu/~smattw/Teaching/cheatsheet445.pdf`.
- Please reference the course collaboration policy here: `http://www.cs.princeton.edu/~smattw/Teaching/infosheet445sp21.pdf`.

[1] We will assign a minor deduction if we need to maneuver around the wrong PDFs. Please also note that depending on if/how you use Overleaf, you may need to recompile your solutions in between downloads to get the right files.

[2] We will assign a minor deduction if we need to maneuver around the wrong PDFs. Please also note that depending on if/how you use Overleaf, you may need to recompile your solutions in between downloads to get the right files.

Problem 1: Super Selfish Mining (40 points)

In this problem, you'll examine a variant of the selfish mining strategy. If it helps, you may think of the following as selfish mining done by a prophet who knows in advance who will be selected to mine in each round.[3] Throughout this problem, you are the attacker, and we will refer to you as m.

Imagine that you control an α fraction of the total computational power in the Bitcoin network, all other miners follow longest-chain and always tie-break against you, and you use the following mining strategy (which is different from lecture). We describe the strategy below in both text and math — it will be helpful to think of the order of operations during every time step as (a) a block is created by a randomly chosen miner, equal to m with probability α and $\neq m$ with probability $1 - \alpha$, (b) all other miners broadcast all of their blocks, and then (c) you may broadcast any blocks you like.

Intuitively, your strategy is exploiting its knowledge of the future, and will only hide a block if you know that you will mine the next block and build a lead of two. Below, note that your decisions on what to do in step t will depend on whether or not you are mining the block in step $t + 1$.

Super Selfish Mining:

- **Notation:** For every time step t,
  - Let M_t denote the identity of the miner who is selected to mine at time t. That is, if $M_t = m$, then the attacker is selected during step t. If $M_t \neq m$, then another miner is selected.
  - At all times t, let $h_m(t)$ denote the height of the highest block mined by you (computed after step (b), before step (c) when you are choosing what to broadcast), and let $h(t)$ denote the height of the highest block that has been broadcast publicly (again after step (b), before step (c)). Recall that the other miners know $h(t)$, but do not know $h_m(t)$ if some of your blocks are hidden.
- **Mining:** During every time step t, mine on top of the longest chain (among all blocks that were broadcast, or created by m), tie-breaking in favor of m (yourself).
- **Broadcasting when $M_t \neq m$:** During every time step t, if another miner broadcasts a block of height $h(t)$ during step (b) and:[4]
  1. $h_m(t) \leq h(t)$, announce your block of height $h(t)$, if one exists (if one does not exist, broadcast nothing).
  2. $h_m(t) > h(t) + 1$, announce your block of height $h(t)$.
  3. $h_m(t) = h(t) + 1$, announce your two blocks of height $h(t)$ and $h(t) + 1$.
- **Broadcasting when $M_t = m$:** During every time step t, if $M_t = m$, decide whether to announce your new block according to the following rule:
  4. If $h_m(t) = h(t) + 1$ **and** $M_{t+1} \neq m$, announce your block of height $h(t) + 1$. That is, if you are mining this round, but not mining next round, and your new block only beats the public chain by one, broadcast your new block immediately.

[3] But if it does not help, you should solve the problem below, exactly as stated, and not focus on this. Note also that you may discover a smarter selfish mining strategy than the one defined below, but you should analyze exactly the strategy defined below, and not something more clever.

[4] Note that this will happen during any round where $M_t \neq m$.

5. If $h_m(t) = h(t) + 1$ **and** $M_{t+1} = m$, **do not announce** any blocks. That is, if you are mining this round and next round, hide your block.
6. If $M_t = m$ and $h_m(t) > h(t) + 1$, **do not announce** any blocks. That is, if you are mining this round and have a large secret lead, hide your block.

Part a (15 points)

Consider the case when miners are selected according to the following sequence (two, six, and seven are not m): $\langle M_1, M_2, M_3, M_4, M_5, M_6, M_7 \rangle = \langle m, \neq m, m, m, m, \neq m, \neq m \rangle$. Which of the six broadcasting cases is triggered during each round? You should present your answer in the form $\langle c_1, c_2, c_3, c_4, c_5, c_6, c_7 \rangle$, where each c_i is an integer from 1 to 6 denoting which case was triggered in round i, followed by a brief justification that each answer is computed correctly.[5]

Part b (25 points)

Describe (or draw, if you prefer) a Markov chain (as a function of α) to analyze the expected reward achieved with the super selfish mining strategy defined above. You should also provide a brief explanation of why your analysis is correct. More specifically, you should provide:

1. A (possibly infinite) list of states.
2. For each pair of states, x and y, the probability of transitioning from state x to y, q_{xy}.[6]
3. For each pair of states, x and y, two values, H_{xy} and S_{xy} (think of H_{xy} as counting the number of blocks that the honest portion of the networks gets in the eventual longest chain when we transition from x to y, and S_{xy} as counting the number of blocks that the selfish miner gets in the eventual longest chain when we transition from x to y).[7]
4. If you find it convenient, you may create multiple "types" of transitions between two states, rather than rigorously add duplicate states. But there are solutions for which this is neither necessary nor convenient. You can ignore this bullet if you find it confusing and/or unhelpful.
5. Your brief justification should clearly prove that every block that is eventually in the longest chain is counted during exactly one transition, and also that only blocks which are in the longest chain are counted.[8]

Your solution should have the property that that if $\vec{p}$ denotes the stationary distribution of your Markov chain (that is, as time goes to infinity, for all x your Markov chain is at state x a p_x fraction of the time), the expected reward achieved by the super selfish mining strategy is:[9]

$$\frac{\sum_{x,y} p_x \cdot q_{xy} \cdot S_{xy}}{\sum_{x,y} p_x \cdot q_{xy} \cdot (S_{xy} + H_{xy})}.$$

[5] If you want to be kind to the graders, you can also type $\boxed{\backslash boxed}$ around your answer.

[6] If for many pairs, $q_{xy} = 0$, you may simply write "all other transition probabilities are zero" after you define the non-zero ones.

[7] Again, you must define all non-zero values, and can declare the rest to be zero.

[8] Your brief justification does not have to be set up to explicitly make both of these statements, but both of these statements should clearly and easily follow from your brief justification.

[9] Observe that if $\vec{p}$ is the stationary distribution, then $p_x \cdot q_{xy}$ is the fraction of time that we spend transitioning from state x to state y.

You do not need to find the stationary probabilities of your Markov chain, nor compute the expected reward achieved by this strategy. You only need to describe the Markov chain as detailed above, and briefly explain why the analysis is correct.

Hint: Refer to Lecture Notes and/or Precept Notes for an example of how to do this for non-super selfish mining.

Problem 2: Obvious Dominance (30 points)

Recall that a complete strategy lists, for that player, what to do at every node where they act. To simplify notation, you may describe a strategy for player one by listing the five actions taken, and a strategy for player two by listing the two actions taken.

Refer to Lecture 23 for a definition of obvious dominance. In the game below, note that every node is in a distinct information set, which greatly simplifies the definition.

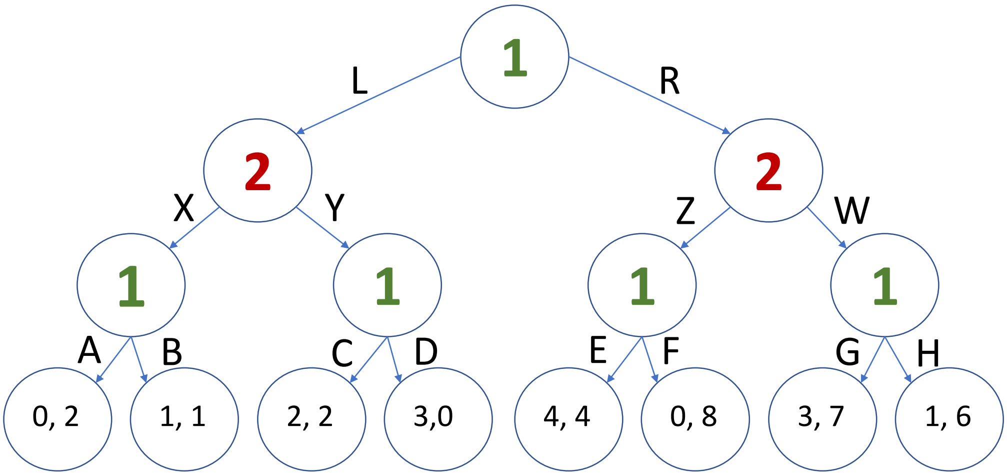

Figure 1: An extended form game.

Part a (15)

Prove that Player One has an Obviously Dominant strategy in the given game.

Part b (15)

Prove that Player Two does not have an Obviously Dominant strategy for the given game.

Problem 3: Bigger Badder Braess' Paradox (30 points)

As a function of $\varepsilon > 0$, describe (or draw, if you prefer) two networks G_ε and G'_ε, and cost functions $c_e(\cdot)$ for each edge, and $\vec{r}$, where r_{ab} units of traffic want to travel from a to b for all nodes a, b, such that:

1. The nodes in G_ε and G'_ε are the same.
2. All edges in G_ε are present in G'_ε, and have the same cost function. **But**, G'_ε may have additional edges. All edge cost functions are continuous, monotone non-decreasing, and non-negative.
3. The unique equilibrium flow in G_ε has total cost at most $1 + \varepsilon$.
4. The unique equilibrium flow in G'_ε has total cost 2.

You should also prove that these properties hold.

Hint: There is a simple solution that looks very similar to the Braess paradox we saw in lecture. To help parse the language in this question, that example itself would resolve the $\varepsilon = 1/2$ case of this question. But you need to resolve this for any $\varepsilon > 0$.

Extra Credit: Proportionality for large n

Let there be n players with normalized, additive values for a cake $[0,1]$. Let also $\mathcal{A}$ denote the set of all partitions of cake to the n players. Let $\mathcal{P}$ denote the set of all proportional partitions of cake to the n players (that is, each player has value at least $1/n$ for their allocated cake).

For notation below, for an allocation $S := S_1 \sqcup \ldots \sqcup S_n$, let $V(S) := \sum_i V_i(S_i)$. Prove that for all n, and all valuations $V_1, \ldots, V_n$, $\max_{S \in \mathcal{A}}\{V(S)\}/\max_{S \in \mathcal{P}}\{V(S)\} = O(\sqrt{n})$. That is, the welfare of the best proportional allocation is at least an $O(\sqrt{n})$ factor of the best welfare without proportional constraints.

For all n, provide a list of valuations $V_1, \ldots, V_n$ such that $\max_{S \in \mathcal{A}}\{V(S)\}/\max_{S \in \mathcal{P}}\{V(S)\} = \Omega(\sqrt{n})$ (that is, prove that the previous bound is tight up to constant factors).

Hint: You will for sure want to use ideas from the algorithm we saw in class to find a proportional allocation.

Hint: Try to break it down into cases where not-that-many players contribute more than $1/\sqrt{n}$ to the total value, and those where many players contribute more than $1/\sqrt{n}$.