Bring Down the ~~Sky~~ Grading Server (grading-server)

by Tobias Lenz and Lucas Schwebler

We write N for the maximum allowed value of c_H, f_H, c_G, f_G. Moreover, we say that a player *sabotages* the other player if they take down one of their firewalls.

Subtask 1. $S, c_H, f_H, c_G, f_G \leq 75$

We can represent every possible state of the game with a tuple of four integers (c_1, f_1, c_2, f_2), where (c_1, f_1) are the computing power and number of firewalls of the current player and (c_2, f_2) are the same for the other player. There are $O(N^4)$ such states, and so we use DP to compute whether each state is a winning or losing state for the first player. This works using the standard observation that a state is winning if and only if it can reach a losing state.

Subtask 2. $S, c_H, f_H, c_G, f_G \leq 300$

Lemma 1. *If (c_1, f_1, c_2, f_2) is winning for player 1, then the state is still winning if we increase c_1 or f_1. In the same way, if the initial state is losing, it stays losing if we increase c_2 or f_2.* □

Thus, we can compute $C(f_1, c_2, f_2)$ as the minimum c_1 such that (c_1, f_1, c_2, f_2) is a winning state. This can be done with DP in $O(N^3 \log N)$ with binary search. It's also possible to compute it in $O(N^3)$ using $C(f_1, c_2, f_2) \leq C(f_1, c_2 + 1, f_2)$.

Implementation detail. One has to be careful when implementing this. Otherwise, the DP might get cyclic dependencies when player 1 attacks. This can be solved by also trying both possible moves for player 2 after an attack by player 1.

Subtask 3. $S = 1$

Define $\alpha_1 := c_1 - S \cdot f_2$ and $\alpha_2 := c_2 - S \cdot f_1$. Intuitively, those values represent the damage caused by a player if they attack the other. Obviously, if $\alpha_1 \leq 0$, the only sensible choice for player 1 is to sabotage. In this interpretation, taking down a firewall just increases α_1 by S and this move can be preformed at most f_2 times. Thus, increasing f_2, for a fixed α_1, makes a state better for player 1, i.e. if $(\alpha_1, f_1, \alpha_2, f_2)$ is a winning state, so is $(\alpha_1, f_1, \alpha_2, f_2 + 1)$.

Lemma 2. *If $\alpha_1 \geq S$ and $\alpha_2 \leq S$, there is a winning strategy for player 1.*

Proof. Player 1 can maintain this invariant by attacking: Afterwards the new value of α_2 is $\alpha_2 - \alpha_1 \leq S - S = 0$. Thus, player 2 has to sabotage. Then, we still have $\alpha_2 \leq S$, and α_1 remains unchanged. □

Lemma 3. *If $\alpha_2 \geq S$, it is optimal for player 1 to attack.*

Proof. Assume for contradiction this were not the case and pick a $(\alpha_1, f_1, \alpha_2, f_2)$ with $\alpha_2 \geq S$ and f_2 minimal, such that sabotaging is a winning move, but attacking is not. In particular, $f_2 \geq 1$.

Suppose that player 1 sabotages, increasing α_1 by S. If player 2 attacks, this will decrease α_1 back to a value $\alpha_1' \leq \alpha_1$. As sabotaging was assumed to be a winning move, the new position $(\alpha_1', f_1, \alpha_2, f_2 - 1)$ is a winning position, hence so is $(\alpha_1, f_1, \alpha_2, f_2 - 1)$. By minimality of our counterexample this means that attacking is a winning move in this position, i.e. $(\alpha_2, f_2 - 1, \alpha_1, f_1)$ is losing. But then also $(\alpha_2, f_2, \alpha_1, f_1)$ has to be losing by the above observation, i.e. attacking was a winning move in the original position. □

Combining those observations with $S = 1$, we can obtain an optimal strategy for player 1. It turns out that whether a state is winning does not depend on f_1, f_2. Let $w(\alpha_1, \alpha_2)$ be **true** if player 1 wins and **false** otherwise. Using the above observations, we can calculate w recursively:

$$w(\alpha_1, \alpha_2) = \begin{cases} \neg w(\alpha_2 - \alpha_1, \alpha_1) & \text{if } \alpha_1, \alpha_2 > 0, \\ \textbf{true} & \text{if } \alpha_1 \geq \alpha_2, \\ \textbf{false} & \text{otherwise.} \end{cases}$$

Evaluating this formula recursively has running time $O(\log N)$ because we repeatedly subtract one value from the other, similar to the Euclidean Algorithm, until one of them becomes ≤ 0.

Subtask 4. $S, c_\text{H}, f_\text{H}, c_\text{G}, f_\text{G} \leq 2\,000$

We can reuse Lemma 2 and 3. This way, we know the optimal strategy if $\alpha_1 \leq 0$, $\alpha_1 \geq S$ or $\alpha_2 \geq S$. Notice that if $f_2 > 0$ and $\alpha_2 \leq 0$, player 1 can sabotage and win with Lemma 2. Thus, the only cases in which we don't know the optimal strategy are those which satisfy

$$0 < \alpha_1, \alpha_2 < S, \quad f_2 > 0, \quad \text{and} \quad f_1, f_2 \leq \frac{N}{S}.$$

The last inequality must hold because otherwise at least one α_i would be negative. We can compute a DP for these unknown states to determine which player wins. There are $O(N^2)$ many such states and each reduction takes $O(\log N)$ by a similar analysis as above.

Subtask 5. $S \leq 400$

Suppose that the game is an interesting state, so we don't know the optimal move yet. If player 1 sabotages, this increases α_1 to a value $> S$. Thus, player 2 has to attack. In total, this increases α_1 by $\beta_2 \coloneqq S - \alpha_2$. Define $\beta_1 \coloneqq S - \alpha_1$ similarly.

Lemma 4 (Death by sabotage). *Player 1 has a winning strategy if $\beta_2 \cdot f_2 \geq \beta_1$.*

Proof. Player 1 sabotages f_2 times in a row. This increases α_1 by $f_2 \cdot \beta_2$. If $\beta_2 \cdot f_2 \geq \beta_1$, then $\alpha_1 + \beta_2 \cdot f_2 \geq S$ and so player 1 wins by Lemma 2. □

Now all interesting states satisfy $\beta_2 \cdot f_2 < \beta_1 < S$. Before attacking, player 1 needs to make sure that player 2 cannot apply the same strategy to win. After the attack of player 1, α_2 is at least $-S$. If $\beta_1 \cdot f_1 \geq 2S$, player 2 can make α_2 larger than S—and therefore win—by taking down all firewalls. So player 1 must not attack if $\beta_1 \cdot f_1 \geq 2S$.

Lemma 5. *There are $O(S^2 \log S)$ interesting states.*

Proof. Using the standard approximation of the harmonic series we compute the number of interesting states as

$$\sum_{\beta_1,\beta_2,f_1,f_2} [\beta_1 f_1 \leq 2S][\beta_2 f_2 \leq \beta_1] \leq \sum_{\beta_1} \sum_{\beta_2} \frac{2S}{\beta_1} \cdot \frac{\beta_1}{\beta_2} = O\left(\sum_{\beta_1} 2S \log \beta_1\right) = O(S^2 \log S).$$ □

Subtask 6. $f_\text{H}, f_\text{G} \leq 125$

We now present a solution which is fast when f_1, f_2 are small.

Lemma 6. *For fixed f_1, f_2 there exists a **critical attack value** $\gamma := \gamma(f_1, f_2)$ such that it is optimal for player 1 to attack if $\alpha_2 \geq \gamma$ and optimal to sabotage otherwise.*

Proof. Consider the states $T_1 = (\alpha_1, f_1, \alpha_2, f_2)$ and $T_2 = (\alpha_1 + 1, f_1, \alpha_2 + 1, f_2)$.

- If attacking is a winning move in T_1, it is also a winning move in T_2: Indeed, after the attack we get the states
$$T_1' = (\alpha_2 - \alpha_1, f_2, \alpha_1, f_1) \quad \text{and} \quad T_2' = (\alpha_2 - \alpha_1, f_2, \alpha_1 + 1, f_1).$$
Those states are identical except for the value of α_1. Since it's higher in T_2', the latter state is still winning for player 1.
- If sabotaging is a losing move in T_1 it is also in T_2: Indeed, after sabotaging and the necessary following attack by player 2, we are in the states
$$T_1' = (\alpha_1 + S - \alpha_2, f_1, \alpha_2, f_2 - 1) \quad \text{and} \quad T_2' = (\alpha_1 + S - \alpha_2, f_1, \alpha_2 + 1, f_2 - 1).$$

For fixed f_1, f_2 let now $h_A(\alpha_2)$ denote the minimum α_1 with which player 1 wins if he attacks in the state $(\alpha_1, f_1, \alpha_2, f_2)$, and let $h_F(\alpha_2)$ be the same if he sabotages. From the above points it follows that
$$h_A(\alpha_2 + 1) \leq h_A(\alpha_2) + 1 \quad \text{and} \quad h_F(\alpha_2 + 1) \geq h_F(\alpha_2) + 1,$$
hence $h_F(\alpha_2 + 1) - h_A(\alpha_2 + 1) \geq h_F(\alpha_2) - h_A(\alpha_2)$. So the difference $\delta(\alpha_2) := h_F(\alpha_2) - h_A(\alpha_2)$ is increasing in α_2. Noticing that attacking is an optimal move for every α_1 if and only if $\delta(\alpha_2) \geq 0$ finishes the proof. □

Observe that once we know all the critical attack values, we can simply simulate the game completely to determine the winner, which in turn allows us to compute all the $\gamma(f_1, f_2)$ recursively. To compute $\gamma(f_1, f_2)$, we use binary search to find minimum α_2 with $h_A(\alpha_2) \leq h_F(\alpha_2)$. Computing h_A, h_F is also done with binary search.

However, to make the simulation efficient enough, we will have to do several sabotages in one step, which requires us to find for a given state the largest $f_2' \leq f_2$ such that attacking is optimal. This can be done using binary search on a segment tree. A somewhat intricate analysis* then reveals that our simulation only takes $O(\log\log S)$ rounds, leading to a runtime of $O\big((Q + F^2 \log^2 F) \log N \log\log S\big)$ where F denotes the maximum value of f_1, f_2.

Subtask 7. No further constraints

The winning strategy from subtask 5 tries to take down all firewalls. Let's try to take down x firewalls, let player 2 do something, and take down the remaining $f_2 - x$ firewalls in the next step. Suppose that the current state is interesting in the sense of subtask 5. If player 1 attacks, player 2 cannot increase the value of α_2 to S (otherwise the state would not be interesting). If player 1 takes down x firewalls before attacking, his new β_1' equals $\beta_1 - \beta_2 x$. Thus, even if player 2 takes down all firewalls, he will have $\alpha_2 \leq S - \beta_2 x f_1$ because every sabotage will increase α_2 by β_2' instead of β_2. This means $\beta_2' \geq \beta_2 x f_1$. Now, player 1 takes down all remaining firewalls, increasing α_1 by at least $\beta_2 f_1 x(f_2 - x)$. If we now specialize to $x := \frac{1}{2} f_2$, this leads to an increase of $\frac{1}{4}\beta_2 f_1 f_2^2$. Since $\alpha_1 > -S$ before player 1 takes down the remaining firewalls, we see that this gives player 1 a winning strategy if
$$\beta_2 f_1 f_2^2 > 8S.$$

Note that in case of $f_1 = 0$ player 1 can still apply the same strategy if $\beta_2 f_2^2 > 8S$, while the case with $f_2 = 0$ is not interesting because player 1 has to sabotage. So all interesting states satisfy $f_2 > 0$ and $\beta_2 \max\{f_1, 1\} f_2^2 \leq 8S$.

* An important step in this analysis is to show that if player 1 attacks, player 2 has to sabotage until $f_1' \leq \frac{f_1}{f_2}$.

Lemma 7. *There are $O(S \log S)$ tuples (f_1, β_2, f_2) with $f_2 > 0$ satisfying $\beta_2 \max\{f_1, 1\} f_2^2 \le 8S$.*

Proof. For any $p \ge 0$, there are

$$\sum_{f_2=1}^{p} \lfloor p / f_2^2 \rfloor \le p \sum_{f_2=1}^{\infty} f_2^{-2} < 2p$$

pairs (f_1, f_2) with $1 \le f_1, f_2$ and $f_1 \cdot f_2^2 \le p$ because the sum $\sum_{i=1}^{\infty} i^{-2}$ converges to $\frac{\pi^2}{6} < 2$. Taking $p = 8S/\beta_2$ and summing over all β_2 then gives the desired bound. □

For every such tuple, we compute the minimum c_1 with which the state is winning with binary search; this gives us an $O(S \log^2 S)$ solution. Again, one needs to implement this carefully to prevent cyclic DP dependencies.

Final remarks.

Combining the solutions for the last two subtasks with some additional ideas, it is also possible to solve this problem in time $O\left(\sqrt{S} \log^4 S + Q\sqrt{S} \log S \log\log S\right)$. Suppose that we are in an interesting state and player 1 attacks. This implies that we had $f_2 < \beta_1$ (otherwise, use death by sabotage). After the attack, player 2 sabotages some rounds until he gets into an interesting state. Then, $f_1^2 f_2^2 < f_1^2 f_2 \beta_1 < 8S$. Notice that there are only $O(\sqrt{S} \log S)$ such pairs (f_1, f_2). Use the idea of subtak 6 for them, leading to a precomputation time of $O(\sqrt{S} \log^4 S)$. To answer a query, we simply try all possible number of sabotages before player 1 attacks. For every of those number of sabotages, we can find the winner very efficiently using the idea from subtask 6. Since $f_2 \le \sqrt{8S}$, the number of sabotage rounds to try is quite small and we get the above time complexity. This is fast enough to solve the problem for constraints $S \le 10^6, Q \le 25\,000$.