Fugitive Frenzy

Problem author and developer: Mikhail Ivanov

Let us call a state of the game *initial* if, from the police officer's perspective, it is effectively same as the beginning of the game in the vertex she is currently in (that is, the police officer does not have any sort of information about the fugitive's position except that he is not sharing a vertex with her). The state before the first move of the game is indeed initial, but also such is every moment when the police officer comes into a leaf (because at that moment all the other vertices form a connected component). Our aim is to prove that, being in the initial state, the police officer should choose at random one of the leaves she is not currently in and walk into it until she gets there and comes into another initial state. As for the fugitive, after an initial state his optimal strategy is to choose at random a leaf unoccupied by the police officer, move there and wait there till the next initial state. Moreover, the probabilistic distribution on these leaves only depends on the accommodation vertex of the police officer in the initial state.

We will prove this in several steps. Firstly, we will assume that at the beginning each of the players generates an infinite random binary string. Each time they need to make a random decision, instead of accessing a random number generator they can just take another one bit (or several ones) from their string. Therefore, we may assume that after the initial string generation their strategies are deterministic. Such pair of strings, generated by the players, will be called an *elementary event*.

We may assume that the fugitive only hides in the leaves of the tree. Indeed, if he choses a non-leaf vertex v, he can as well hide in any leaf reachable from v. It will not change the set of reachable vertices no matter where the police officer moves, but might help him avoid the capture in some elementary events.

Next, we will note that the fugitive may be assumed to not change the shelter after the police officer's step towards him. To prove that, we will change his strategy as follows. Imagine that police officer is in an initial state. Let us fix an elementary event and perform a simulation during which each time the police officer moves, she moves towards the fugitive. At some moment she inevitably catches the fugitive in some leaf ℓ. Note that ℓ can be found deterministically because all randomness exploited by the fugitive is contained within an already generated random string. Then, according to the new strategy, the fugitive prematurely jumps into ℓ and waits till either he is caught or the police officer takes a step away from him. It can be shown that, if the new strategy lead to a capture at some point, the previous strategy also led to a capture no later than the new one elementary event-wise.

In a very similar fashion it can be shown that the police officer should never move along the same edge twice in a row (unless the first of these moves led him into a leaf) — that is, the optimal strategy of the police officer in an initial state consists of choosing some leaf and walking there. Also, with a similar reasoning, it can be shown that the fugitive only needs to chosse his shelter at the beginning of the game and after the police officer getting into a leaf (and thus triggering the initial state).

Therefore, the decisions in this game are only made in initial states. Thus, we can condense the game to its initial states only and assume that the game now is as follows. The police officer appears in vertex s. Then the players take turns, starting with the fugitive. At his turn, the fugitive, who knows the current position of the police officer, chooses a leaf f, different from the police officer's position, and sits there. At the police officer's turn, she chooses a leaf p' and walks there from her current vertex p, spending $\operatorname{dist}(p, p')$ minutes. Then, if $p' \neq f$, the game continues, otherwise, it ends.

In this setting it is easy to see that one move takes at most $n-1$ minutes, and the police officer can guess the correct leaf and finish the game with probability at least $\frac{1}{n-1}$, therefore, the mathematical expectation of the duration of the game, assuming the optimal play by the police officer, is no more than $(n-1)^2$, in particular, it is finite.

Due to Nash, there is a pair of optimal strategies called the *Nash equilibrium*, in which the probabilistic distribution only depends on the position of the police officer. Therefore, we have completed the proof of the fact in question.

Let us fix the optimal strategy. Denote by $p_{u,v}$ the probability that, if the police officer is initially in u, she chooses to walk to v. Denote by $q_{u,v}$ the probability that, if the police officer is initially in u, the

fugitive chooses to hide in v. Denote by x_u the mathematical expectation of the duration of the game if the police officer is initially in u. According to the previous reasoning, $p_{u,v} = q_{u,v} = 0$ unless v is a leaf different from u.

We will prove the converse: both $p_{u,v}$ and $q_{u,v}$ are positive if v is a leaf different from u. To see why, firstly we will assume that $q_{u,v} = 0$ — that is, when the police officer is in u the fugitive never choses to hide in v. Then the police officer, if $p_{u,v} > 0$, can modify her strategy so that she does not visit v, e.g. by stopping right before visiting v and choosing the next leaf right there. This circumstance would violate the Nash equilibrium. Therefore, $q_{u,v} = 0$ implies $p_{u,v} = 0$. Similarly, one can deduce that if $p_{u,v} = 0$ for a leaf $v \neq u$ but $p_{u,v'} > 0$ for some other leaf v' then $q_{u,v'} = 0$: otherwise, the fugitive could modify his strategy and hide in v instead of v', ensuring he doesn't get caught and prolonging the chase. But, as we showed earlier, $q_{u,v'} = 0$ implies $p_{u,v'} = 0$, and that leads to a contradiction. So $p_{u,v}$ cannot equal zero for a leaf $v \neq u$, and so cannot $q_{u,v}$.

So, each of $p_{u,v}$ and $q_{u,v}$ is non-zero if and only if v is a leaf different from u. Our last goal is to calculate the values of $p_{u,v}$, $q_{u,v}$ and x_u. To do that, we will write down several equations on these numbers. Denote by L the set of leaves of the tree. Firstly, x_u, as the mathematical expectation of the duration of the chase, satisfies:

$$x_u = \sum_{v \in L \setminus \{u\}} F(u,v) \qquad \text{where } F(u,v) = p_{u,v}(\operatorname{dist}(u,v) + (1 - q_{u,v})x_v).$$

Also, as a solution to an optimization problem, $p_{u,v}$ and $q_{u,v}$ satisfy *complementary slackness conditions* (as a part of Karush–Kuhn–Tucker conditions). Namely, note that if, for a fixed u, $P_u(v) = \frac{\partial}{\partial p_{u,v}} F(u,v) = (\operatorname{dist}(u,v) + (1 - q_{u,v})x_v)$ differs for two different vertices $v \in L \setminus \{u\}$, then it would be profitable for the police officer to always walk into the vertex with the lower value of $P_u(v)$, and that would violate the Nash equilibrium. Similarly, for a fixed u, $Q_u(v) = \frac{\partial}{\partial q_{u,v}} F(u,v) = -p_{u,v}x_v$ should be equal among all $v \in L \setminus \{u\}$, otherwise the fugitive would have the incentive to only hide in the vertices v with the maximum $Q_u(v)$ and violate the Nash equilibrium from his side.

The aforementioned complementary slackness conditions can be reformulated in the following way: if in the function $F(u,v) = p_{u,v}(\operatorname{dist}(u,v) + (1 - q_{u,v})x_v)$ one replaces the array $p_{u,v}$ with any other array with a unit sum, which vanishes on $v \notin L \setminus \{u\}$, then the value of $F(u,v)$ does not change. Similarly, if one replaces the array $q_{u,v}$ with any other array with a unit sum, which vanishes on $v \notin L \setminus \{u\}$, then the value of $F(u,v)$ does not change. (But if one does the both modifications, then $F(u,v)$ *might* change.)

Now we are ready to write a system of equations on the numbers x_u. All numbers $p_{u,v}x_v$ are equal to each other, so, for a fixed $v \in L \setminus \{u\}$,

$$1 = \sum_{w \in L \setminus \{u\}} p_{u,w} = \sum_{w \in L \setminus \{u\}} p_{u,w}x_w \cdot \frac{1}{x_w} = \sum_{w \in L \setminus \{u\}} p_{u,v}x_v \cdot \frac{1}{x_w} = p_{u,v}x_v \cdot \sum_{w \in L \setminus \{u\}} \frac{1}{x_w},$$

therefore,

$$p_{u,v} = \frac{1/x_v}{\sum_{w \in L \setminus \{u\}} 1/x_w}.$$

Let us substitute this expression in the formula for $F(u,v)$:

$$F(u,v) = \frac{1/x_v}{\sum_{w \in L \setminus \{u\}} 1/x_w}(\operatorname{dist}(u,v) + (1 - q_{u,v})x_v).$$

It was said earlier that the array $q_{u,v}$ can be chosen arbitrarily if the unit sum is preserved. We will fix a vertex $t \in L \setminus \{u\}$ and take $q_{u,v} = [v = t]$ — that is, $q_{u,t} = 1$ and the rest of the $q_{u,v}$ are taken zero. After this substitution, $F(u,v)$ stays the same:

$$F(u,v) = \frac{1/x_v}{\sum_{w \in L \setminus \{u\}} 1/x_w}(\operatorname{dist}(u,v) + x_v - [v = t]x_t).$$

The formula for x_u takes the form:

$$x_u = \sum_{v \in L \setminus \{u\}} \frac{1/x_v}{\sum_{w \in L \setminus \{u\}} 1/x_w}(\operatorname{dist}(u,v) + x_v - [v = t]x_v).$$

The $x_v - [v = t]x_v$ part can be removed from under the summation sign:

$$x_u = \frac{|L \setminus \{u\}|}{\sum_{w \in L \setminus \{u\}} 1/x_w} - \frac{1}{\sum_{w \in L \setminus \{u\}} 1/x_w} + \sum_{v \in L \setminus \{u\}} \frac{1/x_v}{\sum_{w \in L \setminus \{u\}} 1/x_w} \operatorname{dist}(u, v).$$

This already yields a rather convenient expression:

$$x_u = \frac{|L| - 2 + [u \notin L] + \sum_{v \in L \setminus \{u\}} \operatorname{dist}(u, v)/x_v}{\sum_{v \in L \setminus \{u\}} 1/x_v}.$$

However, to get a bit more symmetric form, one can multiply it by the denominator:

$$x_u \cdot \sum_{v \in L \setminus \{u\}} 1/x_v = |L| - 2 + [u \notin L] + \sum_{v \in L \setminus \{u\}} \operatorname{dist}(u, v)/x_v.$$

Then, if $u \in L$, we can add one to both sides (or, equivalently, add $[u \in L]$ to both sides):

$$x_u \cdot \sum_{v \in L} 1/x_v = |L| - 1 + \sum_{v \in L} \operatorname{dist}(u, v)/x_v.$$

Finally, let us divide it back:

$$x_u = \frac{|L| - 1 + \sum_{v \in L} \operatorname{dist}(u, v)/x_v}{\sum_{v \in L} 1/x_v}.$$

Unfortunately, this system of equations is very unlikely to have a clear way to be solved analytically in general case. Even for small trees the exact formulae begin to look gargantuan, and there are really few classes of trees with neat formulae (e.g. star, bamboo, pair of equal stars connected by a bamboo). Instead, one can take some initial approximation, e.g. $x_u = 1$ or $x_u = (n-1)^2$, and iteratively apply the formula above to get a more and more precise answer. This process seems to converge relatively fast, although jury does not have a proof of that. What is worth noting is that, at first, it is reasonable to only calculate x_u for $u \in L$ (since these are the only x_u which occur in the right hand side), and in the end, when the leaf expectations are already calculated with satisfying precision, calculate the rest of the expectations with one iteration over all $u \in \overline{L}$.

One more possible speedup is to work with $y_u = 1/x_u$ instead of x_u, since this would lead to a much smaller number of divisions:

$$y_u = \frac{\sum_{v \in L} y_v}{|L| - 1 + \sum_{v \in L} \operatorname{dist}(u, v) y_v}.$$

This final approach works in $\mathcal{O}\big(n\ell + q\ell^2 + n\ell\big) = \mathcal{O}\big(n\ell + q\ell^2\big)$, where $n = |V|$, $\ell = |L|$ and q is the number of iterations: the first $n\ell$ summand stands for finding the pairwise distances between the vertices and the leaves, $q\ell^2$ stands for the iterative algorithm for finding the leaf values of x_u, and the last $n\ell$ summand stands for calculating the rest of the values of x_u. In practice, for $n \le 100$, $q = 1500$ or $q = 2000$ and a 64-bit floating point number type were enough to pass all tests, depending on the quality of the initial approximation. The time limit for this problem is pretty loose, so even Python solutions should pass with a reasonable implementation.