Then $\text{fp}'(G^S, 0) = \sum_{u_i \in S} \alpha(u_i)$.

The intuition behind the quantities π_i, ψ_{ij}, and $\alpha(u_i)$ from Lemma 7 is as follows: Consider the Moran process on G with $\delta = 0$. Then for $i \in [n]$, the value π_i is in fact the mutant fixation probability starting from the initial configuration $\mathsf{X}_0 = \{u_i\}$ (**?**). Indeed, at any time step, the agent on u_i will eventually fixate if (i) u_i is not replaced by its neighbors and eventually fixates from the next step onward, or (ii) u_i spreads to a neighbor u_j and fixates from there. The first event happens with rate $(1 - \sum_{j \in [n]} p_{ji}) \cdot \pi_i$, while the second event happens with rate $\sum_{j \in [n]} p_{ij} \cdot \pi_j$. (We note that for undirected graphs, the system has an explicit solution $\pi_i = \frac{1/\deg(u_i)}{\sum_{j \in [n]} 1/\deg(u_j)}$ (**?**).)

The values ψ_{ij} for $i \neq j$ also have intuitive meaning: They are equal to the expected total number of steps during the Moran process (with $\delta = 0$) in which u_i is mutant and u_j is not, when starting from a random initial configuration $\mathbb{P}[\mathsf{X}_0 = \{u_i\}] = 1/n$. To sketch the idea behind this claim, suppose that in a single step, a node u_ℓ spreads to node u_i (this happens with rate $p_{\ell i}$). Then the event "u_i is mutant while u_j is not" holds if and only if u_ℓ was mutant but u_j was not, hence the product $p_{\ell i} \cdot \psi_{\ell j}$. Similar reasoning applies to the product $p_{\ell j} \cdot \psi_{i \ell}$. The term 1 in the numerator comes out of the $1/n$ probability that the initial mutant lands on u_i (in which case indeed u_i is mutant and u_j is not).

Finally, the function $\alpha(\cdot)$ also has a natural interpretation: The contribution of an active node u_i to the fixation probability grows vs. δ at $\delta = 0$ to the extent that a mutant u_i is likely to be chosen for reproduction, spread to a resident neighbor u_j, and fixate from there; that is, by the total time that u_i is mutant but a neighbor u_j is not (ψ_{ij}), calibrated by the probability that, when chosen for reproduction, u_i propagates to that neighbor (p_{ij}), who thereafter achieves mutant fixation (π_j); this growth rate is summed over all neighbors and weighted by the rate $1/n$ at which u_i reproduces when $\delta = 0$.

5 Experiments

Here we report on an experimental evaluation of the proposed algorithms and some other heuristics. Our data set consists of 110 connected subgraphs of community graphs and social networks of the Stanford Network Analysis Project (**?**). These subgraphs where chosen randomly, and varied in size between 20-170 nodes. Our evaluation is not aimed to be exhaustive, but rather to outline the practical performance of various heuristics, sometimes in relation to their theoretical guarantees.

In particular, we use 7 heuristics, each taking as input a graph, a budget k, and (optionally) the mutant fitness advantage δ. The heuristics are motivated by our theoretical results and by the related fields of influence maximization and evolutionary graph theory.

1. *Random:* Simply choose k nodes uniformly at random. This heuristic serves as a baseline.
2. *High Degree:* Choose the k nodes with the largest degree.
3. *Centrality:* Choose the k nodes with the largest betweenness centrality.
4. *Temperature:* The temperature of a node u is defined as $\mathcal{T}(u) = \sum_v w(v, u)$; this heuristic chooses the k nodes with the largest temperature.
5. *Vertex Cover:* Motivated by Lemma 4, this heuristic attempts to maximize the number of edges with at least one endpoint in S. In particular, given a set $A \subseteq V$, let
$$c(A) = \{|(u, v) \in E \colon u \in A \text{ or } v \in A|\}. \quad (13)$$
The heuristic greedily maximizes $c(\cdot)$, i.e., we start with $S = \emptyset$ and perform k update steps
$$S \leftarrow S \cup \arg\max_{u \notin S} c(S \cup \{u\}). \quad (14)$$
6. *Weak Selector:* Choose the k nodes that maximize $\text{fp}'(G^S, 0)$ using the (optimal) weak-selection method.
7. *Lazy Greedy:* A simple greedy algorithm starts with $S = \emptyset$, and in each step chooses the node u to add to S that maximizes the objective function. For $\texttt{FM}(G, \delta, k)$ and $\texttt{FM}^\infty(G, k)$ this process requires to repeatedly evaluate $\text{fp}(G^{S \cup \{u\}}, \delta)$ (or $\text{fp}(G^{S \cup \{u\}}, \delta)$) for every node u. This is done by simulating the process a large number of times (recall Lemma 5), and becomes a computational bottleneck when we require high precision. As a workaround we suggest a lazy variant of the greedy algorithm (**?**), which is faster but requires submodularity of the objective function. In effect, this algorithm is a correct implementation of the greedy heuristic in the limit of strong selection (recall Lemma 6), while it may still perform well (but without guarantees) for finite δ.

In all cases ties are broken arbitrarily. Note that the heuristics vary in the amount of information they have about the graph and the invasion process. In particular, Random has no information whatsoever, High Degree only considers the direct neighbors of a node, Temperature considers direct and distance-2 neighbors of a node, Centrality and Vertex Cover consider the whole graph, while Weak Selector and Lazy Greedy are the only ones informed about the Moran process.

For each graph G, we have chosen values of k corresponding to 10%, 30% and 50% of its nodes, and have evaluated the above heuristics in their ability to solve $\texttt{FM}^\infty(G, K)$ (strong selection) and $\texttt{FM}^0(G, k)$ (weak selection). We have not considered other values of δ as evaluating $\text{fp}(G^S, \delta)$ precisely via simulations requires many repetitions and becomes slow.

Strong selection. We start with the case of $\texttt{FM}^\infty(G, K)$. Since different graphs G and budgets k generally result in highly variant fixation probabilities, in order to get an informative aggregate measure of each heuristic, we divide the fixation probability it obtains by the maximum fixation probability obtained across all heuristics for the same graph and budget. This normalization yields values in the interval $[0, 1]$, and makes comparison straightforward. Fig. 4 shows our results. We see that the Lazy Greedy algorithm performs best for small budgets (10%, 30%), while its performance is matched by Vertex Cover for budget 50%. The high performance of Lazy Greedy is expected given its theoretical guarantee (Theorem 3). We also observe that, apart from Weak Selector, the other heuristics perform quite well on many

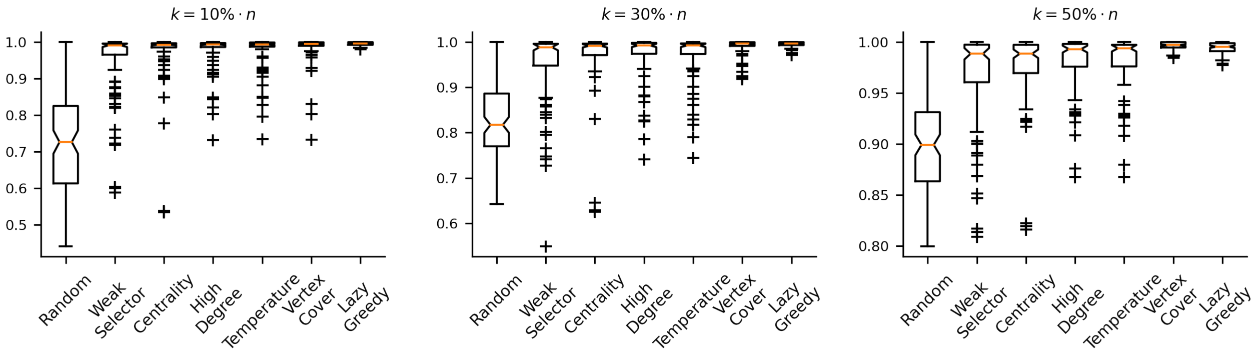

Figure 4: Heuristic performance for $\texttt{FM}^{\infty}$.

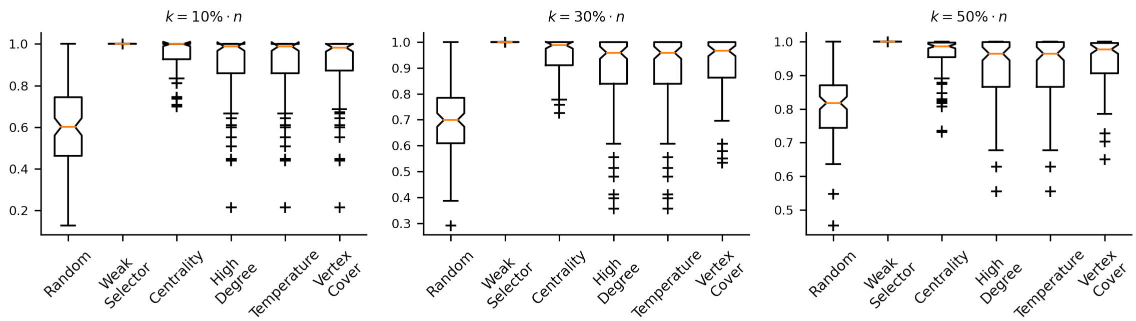

Figure 5: Heuristic performance for $\texttt{FM}^{0}$.

graphs, though there are several cases on which they fail, appearing as outlier points in the box plots. As expected, our baseline Random heuristic performs quite poorly; this result indicates that the node activation set S has a significant impact on the fixation probability. Finally, recall that Weak Selector is optimal for the regime of weak selection ($\delta \to 0$). The fact that Weak Selector underperforms for strong selection ($\delta \to \infty$) indicates an intricate relationship between fitness advantage and fixation probability.

Weak selection. We collect the results on weak selection in Fig. 5, using the normalization process above. Since the derivative satisfies $\mathrm{fp}'(G^S, 0) = \sum_{u_i \in S} \alpha(u_i)$ (Section 4.3), Lazy Greedy is optimal and coincides with Weak Selector, and is thus omitted from the figure. Naturally, the Weak Selector always outperforms others, while the Random heuristic is weak. The other heuristics have mediocre performance, with a clear advantage of Centrality over the rest, which becomes clearer for larger budget values.

6 Conclusion

We introduced the positional Moran process and studied the associated fixation maximization problem. We have shown that the problem is NP-hard in general, but becomes tractable in the limits of strong and weak selection. Our results only scratch the surface of this new process, as several interesting questions are open, such as: Does the strong-selection setting admit a better approximation than the one based on submodularity? Can the problem for finite δ be approximated within some constant-factor? Are there classes of graphs for which it becomes tractable?

Voluptatum dolores distinctio iusto commodi reiciendis a nobis, commodi ut magni cumque repellat facilis molestiae illum deserunt hic rem sint.Aliquid deleniti enim, deleniti possimus adipisci maiores voluptatem, soluta ad nemo itaque praesentium perferendis temporibus illum fugiat?Nulla nemo itaque necessitatibus, tempora quae earum doloribus minima aut quos pariatur incidunt, nemo nostrum facilis culpa totam ab cum, optio nulla cupiditate.Aperiam sint sed ex dicta magni soluta eius, reprehenderit minus veniam fugiat, voluptatibus repudiandae vel, eveniet nihil et aliquid dolorem totam ipsa?Voluptatem consequuntur numquam omnis fugit similique ullam, officia maxime reiciendis labore ut accusamus ipsum consequuntur quae, quae harum possimus deleniti ad sapiente odio dolor pariatur molestiae.Expedita ab officiis ex ipsam perferendis temporibus dolorum optio voluptate, incidunt inventore corrupti quam fuga illum iste numquam aliquid sequi, iure quae illo reiciendis eius soluta quaerat animi rem dolorum, voluptate quo nihil minima harum quibusdam quas, possimus tempora nisi ab quod voluptatum?Earum quae dolorum neque assumenda aut eaque cum asperiores ipsa itaque ut, molestiae ratione dolores nesciunt at ipsa deleniti.Nemo in ipsa quis cum est, minus iste nesciunt, maiores magni est dolor excepturi?Fugit quod eligendi delectus, dicta voluptates autem minus?