MONASH University

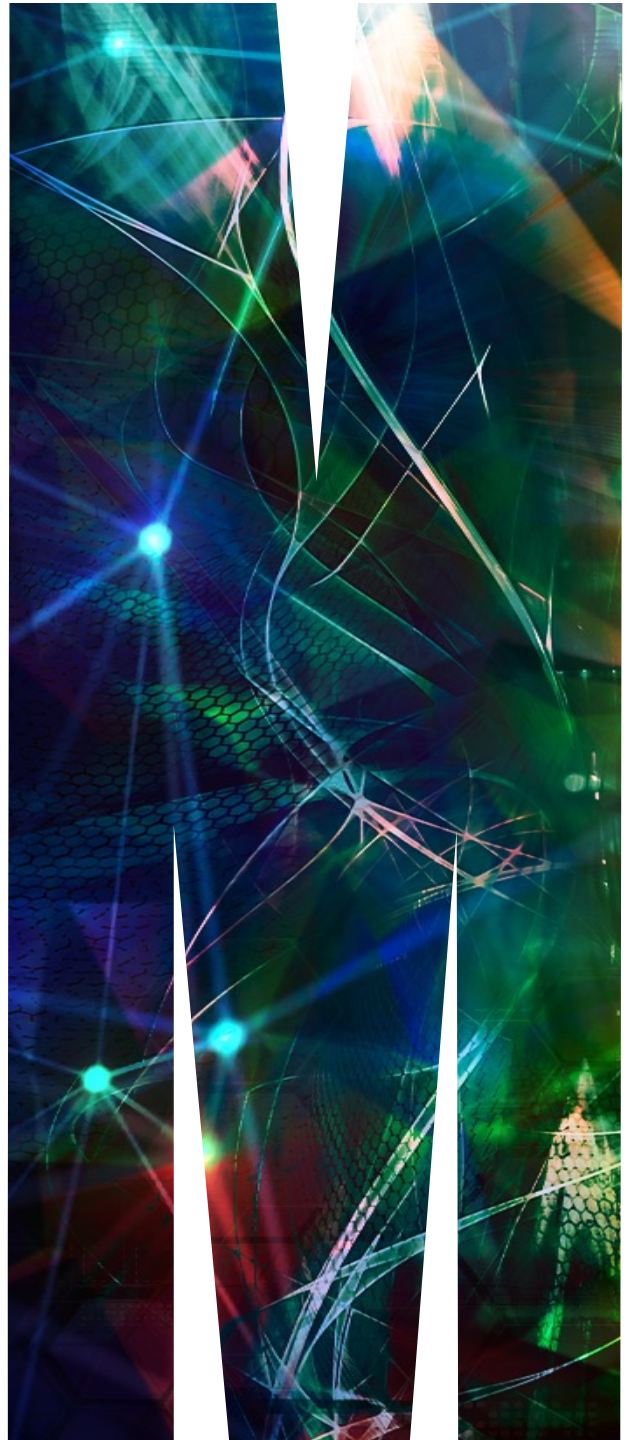
Search-based optimisation and social fuzz testing

A/Prof. Markus Wagner

http://acrocon.com/~wagner/

14 March 2023

Clayton, VIC



Acknowledgement of Country

Monash University recognises that its Australian campuses are located on the unceded lands of the people of the Kulin Nations, and pays its respects to their Elders, past, present, and emerging.

MONASH University

About Markus

*** Looking for collaborations ***

Joined Monash University in January ’23

`Monash.FIT.DSAI.OptimisationGroup.members.containsKey(this) = true`

DSAI Director of Industry Engagement

FIT Sustainable Energy Informatics Theme Lead (RACE for 2030 CRC)

PhD in ‘13, University of Adelaide

“Theory and Applications of Bio-Inspired Algorithms”

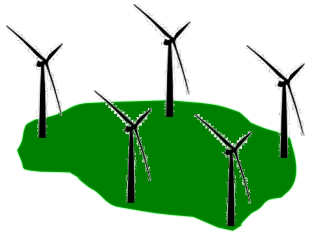
Achievements

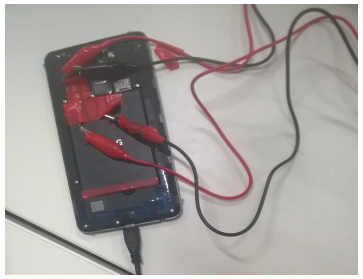
150 papers, 200 co-authors, h-index 35, 3.5k citations,
AUD 10M (AUD 1.6M as lead),
3 Best Papers, 1 Best Presentation, 1 Best Poster, 1 Medal,
ACM: GECCO General Chair, SIGEVO Sustainability Officer, …
IEEE: CIS Task Force founder (2x), CIS High School Outreach Chair, …

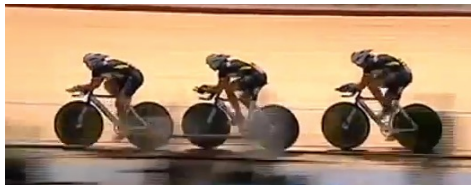
Current team

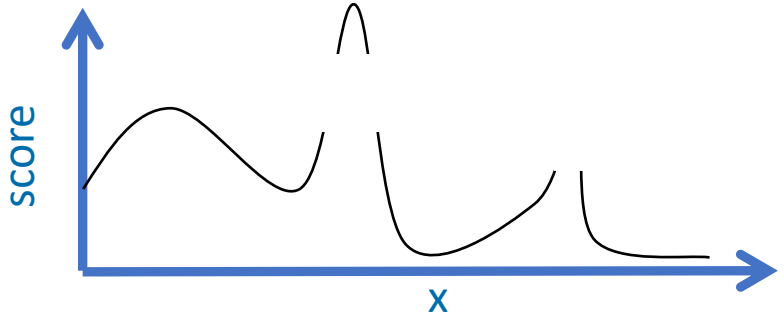
3 FIT3144, 2 MPhils, 4 PhDs, 1 RA

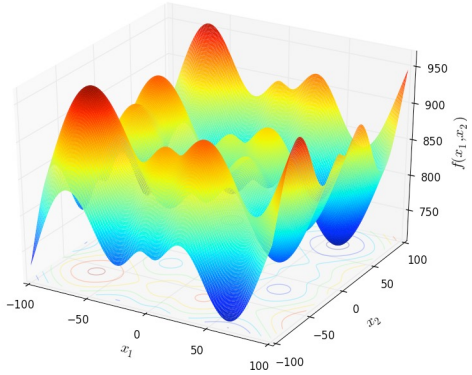
Real-world optimisation











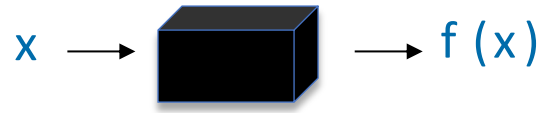
Problem-specific (or exact mathematical) algorithms not always available

… problem is not entirely understood

… objective function is based on a simulation

… lot of non-linear pieces

… not enough resources



Heuristic Optimisation – my field

Heuristic Approaches

… do not rely on gradient information.

… are less likely to get stuck due to inherent parallelism.

General Template

1. Choose a representation for the potential solutions.
2. Choose a function to evaluate the quality.
3. Define operators that produce new solutions.

Examples: Grid Search, Local Search, Variable Neighbourhood Search, Simulated Annealing, Evolutionary Algorithms, Ant-Colony Optimisation, …

Can it get funded?

"Dynamic Adaptive Software Configurations" (ARC)
"Intelligent Technologies for Smart Cryptography" (ARC)
"Automatic Post-Quantum Cryptographic Code Generation and Optimization" (Google)
"Rewriting software documentation for non-native speakers" (Google)
"Contextually Situated Anomaly Detection" (Defence Innovation Partnership)
"Collaborative Sensing and Learning for Maritime Situational Awareness" (ARC)
"Socialz – Multi-Objective Automated Social Fuzz Testing" (Facebook) *…coming up in 5 min…*

Algorithmic Interests

1) Achieve the best results

Related: how to benchmark? Check out “Benchmarking in Optimization: Best Practice and Open Issues”

2) Reach a local optimum quickly

self-adaptive parameter control (based on feedback during the search); heavy-tailed probabilities (instead of the sharply concentrated 1/n); use restarts (“bet-and-run”)

3) Problem types

Optimisation under noise (phones are the worst: unreliable software, sensor drift, system states, …)
Multi-objective optimisation
Problems with multiple interdependent components (Decompose? Multiple levels?)

4) Learn about the problem

Systematic analyses of local optima: visualise the neighbourhood; local optima networks, and then feed that knowledge back into the design of the variation operators*

Use “Automatic algorithm configuration” (=tuning) to learn about set of problem instances

MSR/EMSE papers with Tim Menzies on the general ideology of always using optimisation):

“Better Software Analytics via DUO: Data Mining Algorithms Using/Used-by Optimizers” (→ fine tune yourself and your competitors)
“Data-Driven Search-based Software Engineering”

5) Compute diverse solutions/entities

structurally diverse TSP instances OR discriminating TSP instances (e.g. for benchmarking purposes, for model-building purposes, …)
Images that are different w.r.t. features (but not too far away from the original)
Diversified community interaction in a social networks

Let's finally get to a research project…

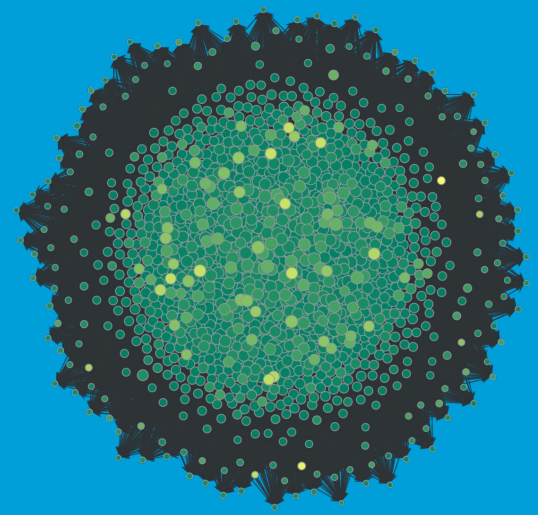
Socialz: Multi-Feature Social Fuzz Testing

In Collaboration with
Francisco Zanartu (UAdelaide) and
Christoph Treude (UMelbourne)

https://arxiv.org/abs/2302.08664
https://github.com/fzanart/Socialz
under review

MONASH University

Supported by Facebook/Meta



* Includes two algorithmic building blocks

Bugs in Online Social Networks

ars TECHNICA BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE

TRUTHS AND RETRUTHS —

Trump's social app marred by bugs and apparent ban on Devin Nunes cow accounts

Trump's social network technically exists now, but good luck trying to use it.

JON BRODKIN - 2/24/2022, 7:49 AM

BUSINESSPLUS

SCIENCE & TECHNOLOGY / 5TH NOVEMBER 2022 / GEORGE MORAHAN

Data breaches have become a hazard of being on social media, but some websites are worse at handling our data than others.

In Ireland alone, the Data Protection Commission received notifications of 6,549 data breaches last year and issued a fine of €225m to Meta-owned WhatsApp over a range of compliance failures.

Affected users:
Yahoo: 3.5bns
Facebook: 2.1bn
LinkedIn: 1.1bn
MySpace: 0.7bn
Sina Weibo: 0.5bn
…

Social Bugs

Social bugs are the result of community interaction, see “WES: Agent-based user interaction simulation on real infrastructure”.

Social testing poses challenges: data collection (time-consuming, difficult, illegal), setup cost.

Socialz to the rescue!

A novel approach to social fuzz testing that

(1) characterises real users of a social network,

(2) diversifies their interaction using evolutionary computation across multiple, non-trivial features, and

(3) collects performance data as these interactions are executed.

With Socialz, we aim to provide anyone with the capability to perform comprehensive social testing, thereby improving the reliability and security of online social networks used around the world.

Socialz – Overview

Target platform

Three-stage approach for fuzz testing OSNs

- 1/3: Characterisation of User Behaviour
- 2/3: Evolutionary Diversification of Community Interaction
- 3/3: Execution

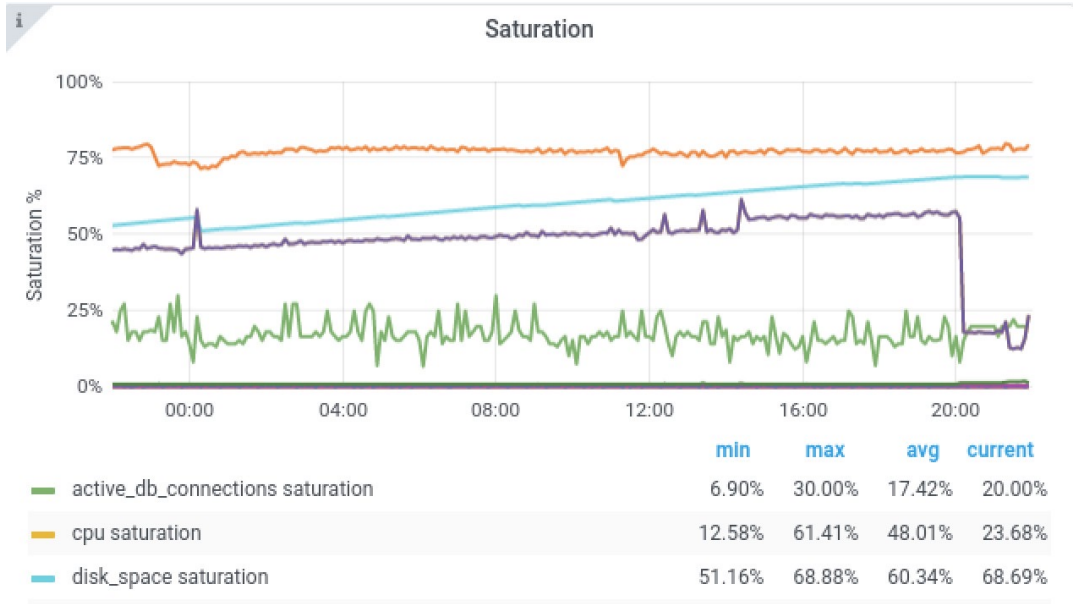
Efficient evolution and evaluation*

Conclusions and future work (and *then* the actual Conclusion of this seminar)

Target Platform: GitLab Community Edition

Useful features

- GitLab is widely used (>30M users)
- GitLab CE is free and open source
- Provides comprehensive set of performance metrics
- Prometheus time-series database
- Pre-defined Grafana dashboards
- Docker container



Stage 1/3: Characterisation of User Behaviour

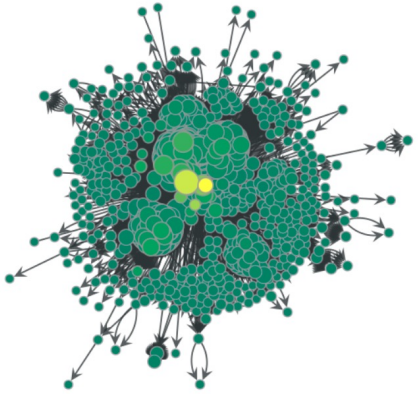
Based on GitHub Archive.

Original dataset: 1,523 users created a total of 6,742 events involving 156 repositories and forks (2011–2016). Subcommunity: COBOL.

Event type	Number of events	
PushEvent	4234	≈ being invited to a group with permission to publish some content
WatchEvent	1206	≈ liking a public profile page
PullRequestEvent	852	≈ requesting permission to publish something to a group
ForkEvent	450	≈ liking a public profile page
FollowEvents*		≈ establishing a connection with another user

Stage 2/3: Evolutionary Diversification of Interaction

The “Original” dataset



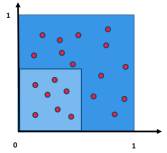
Our ambition: non-trivial features of Community Interaction…

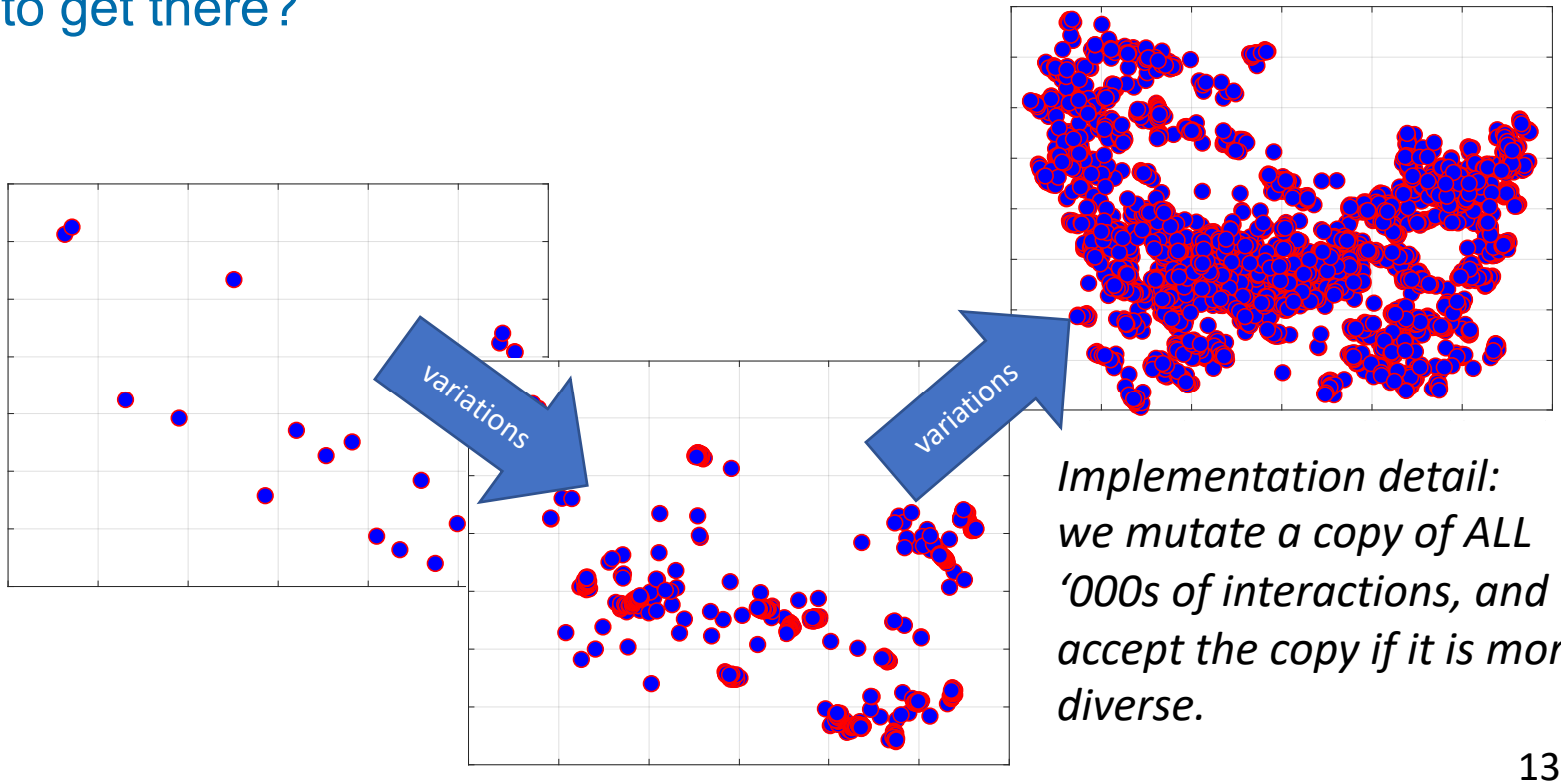
- Degree of Centrality (number of edges)
- PageRank (“importance”)
- Event types (16* different combinations)

How to get there?

Evolutionary Diversity Optimisation!

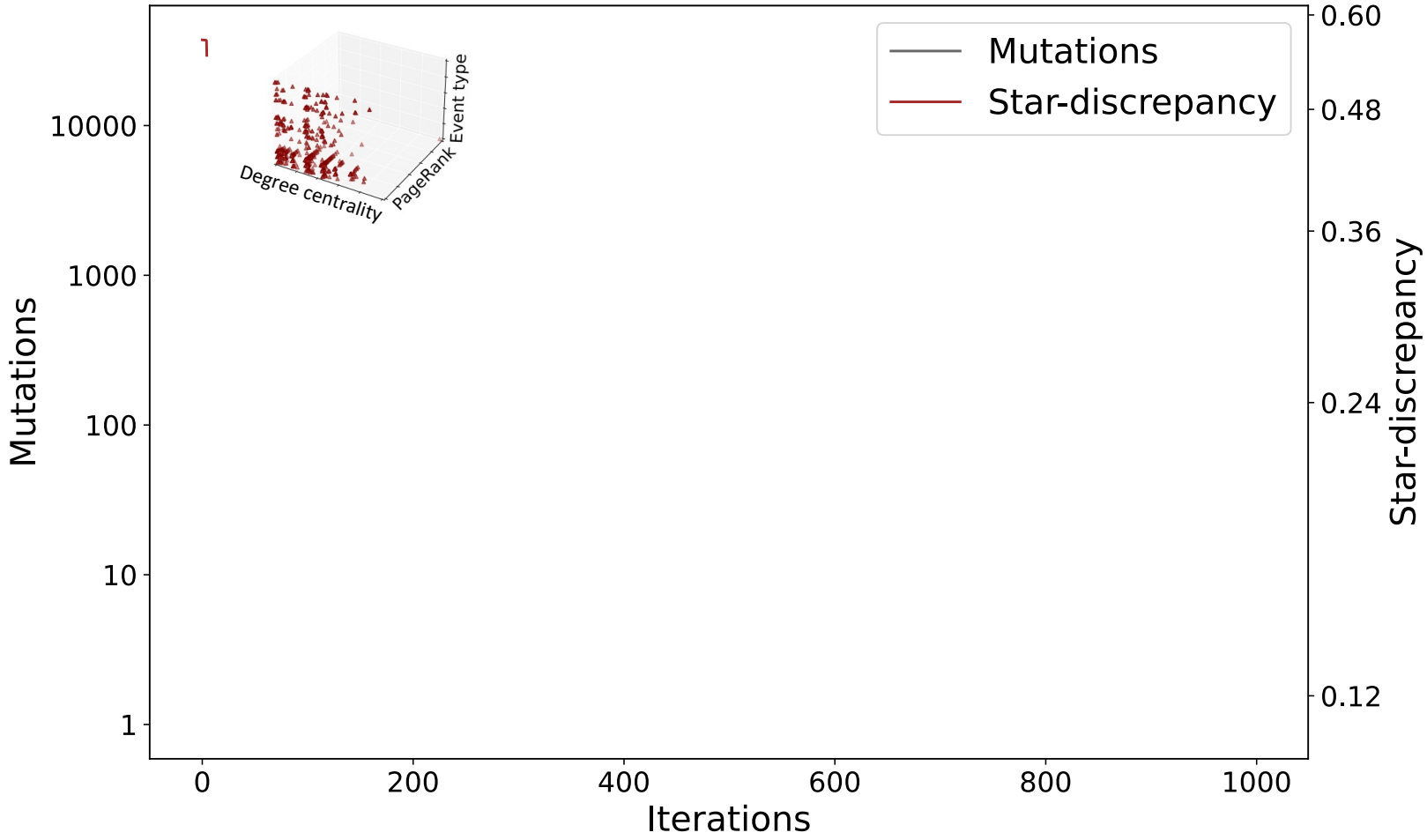
Imagine the following:
1) The plot on the left, we consider 20 users in the feature space.
2) We use ”star discrepancy” as a measure to tell us how evenly the points are distributed.



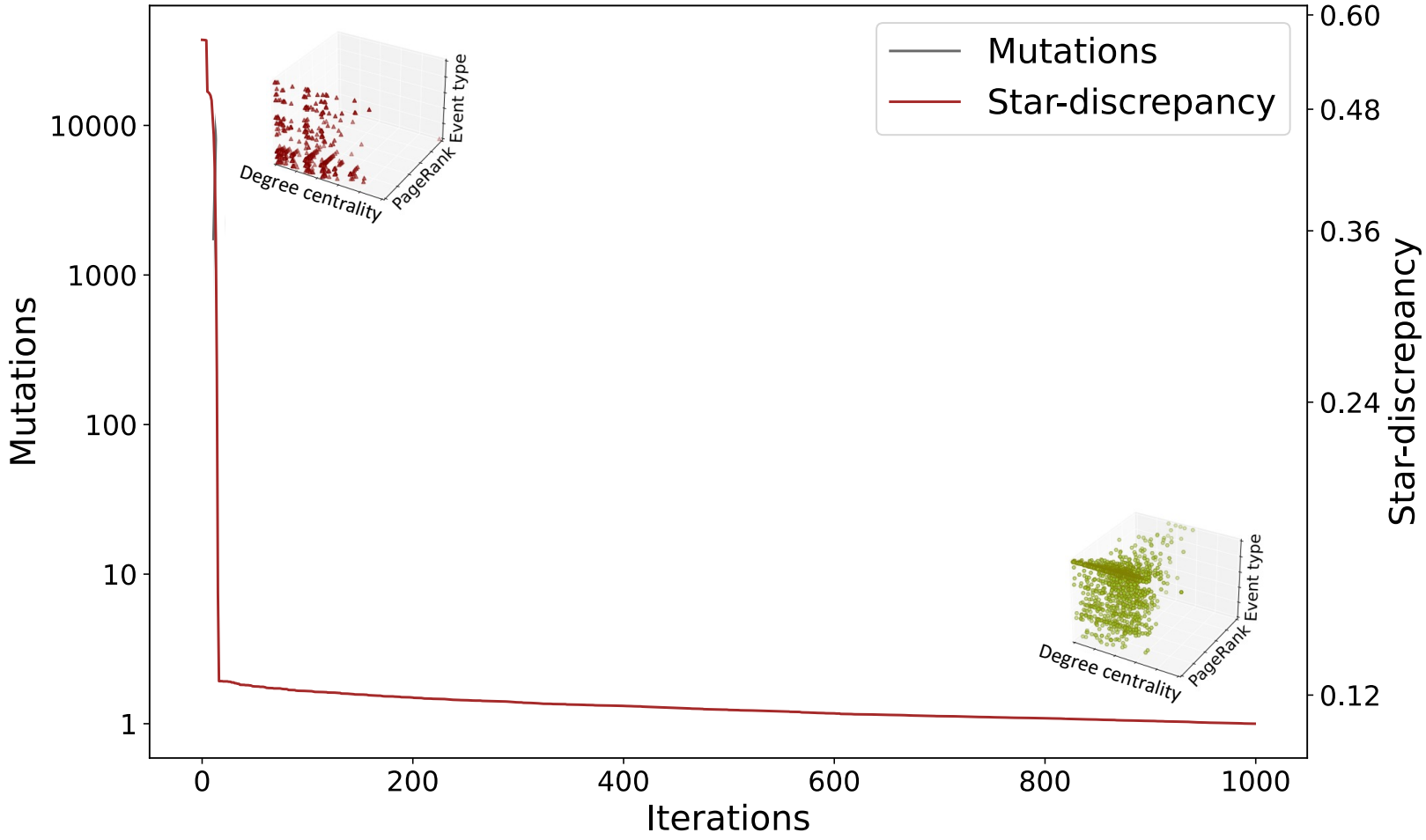


Implementation detail: we mutate a copy of ALL ‘000s of interactions, and accept the copy if it is more diverse.

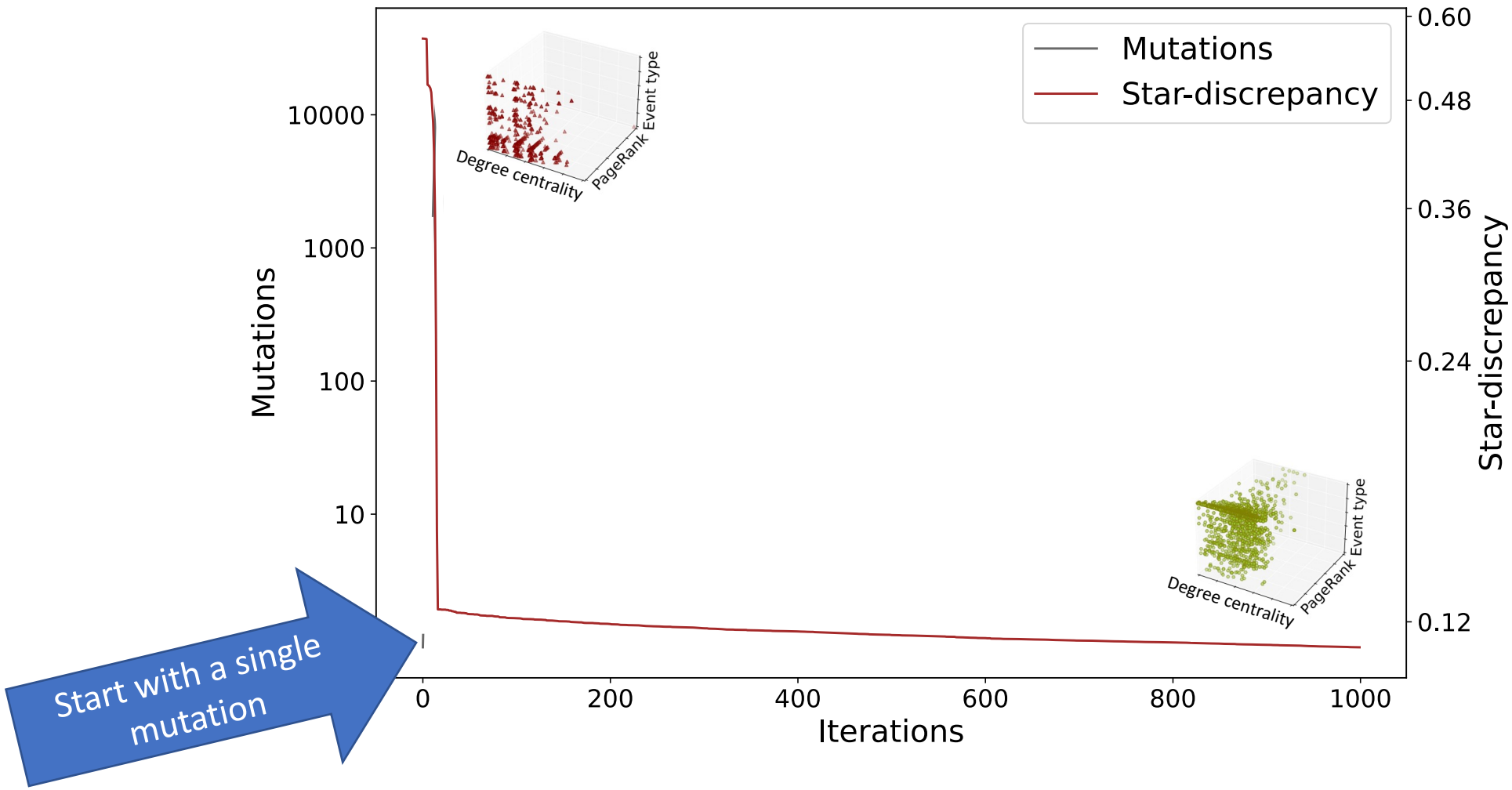
Stage 2/3: Evolutionary Diversification of Interaction



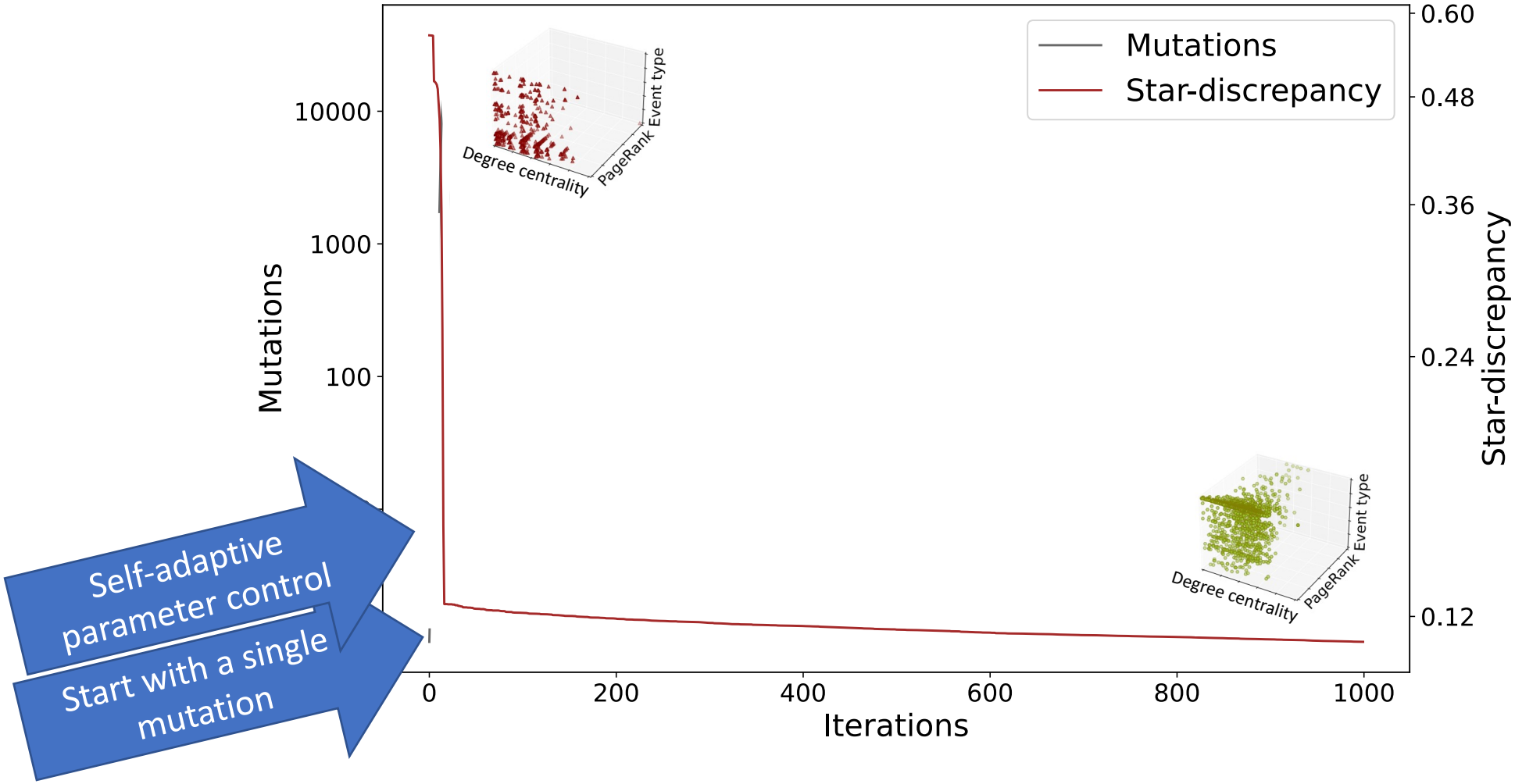
Stage 2/3: Evolutionary Diversification of Interaction



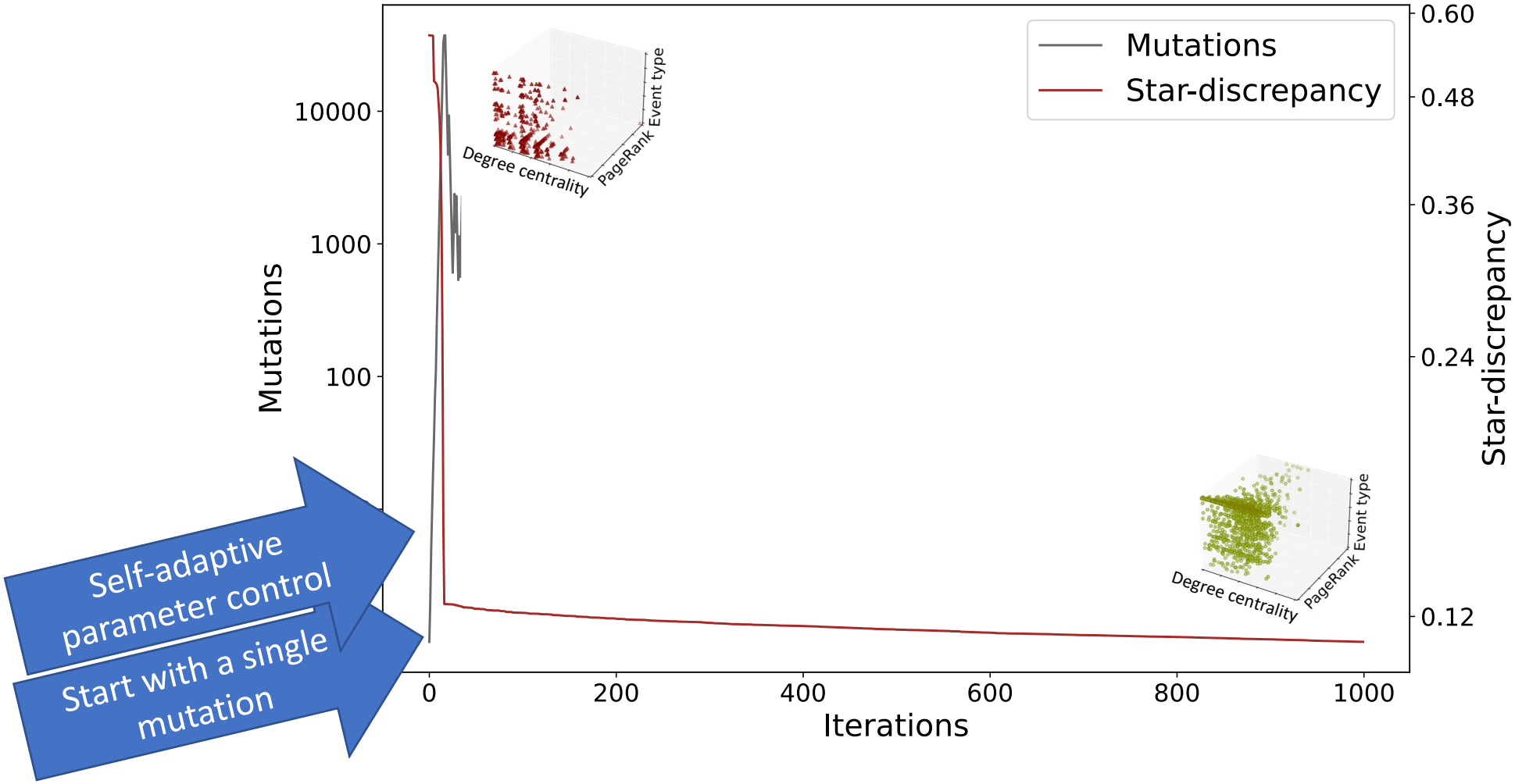
Stage 2/3: Evolutionary Diversification of Interaction



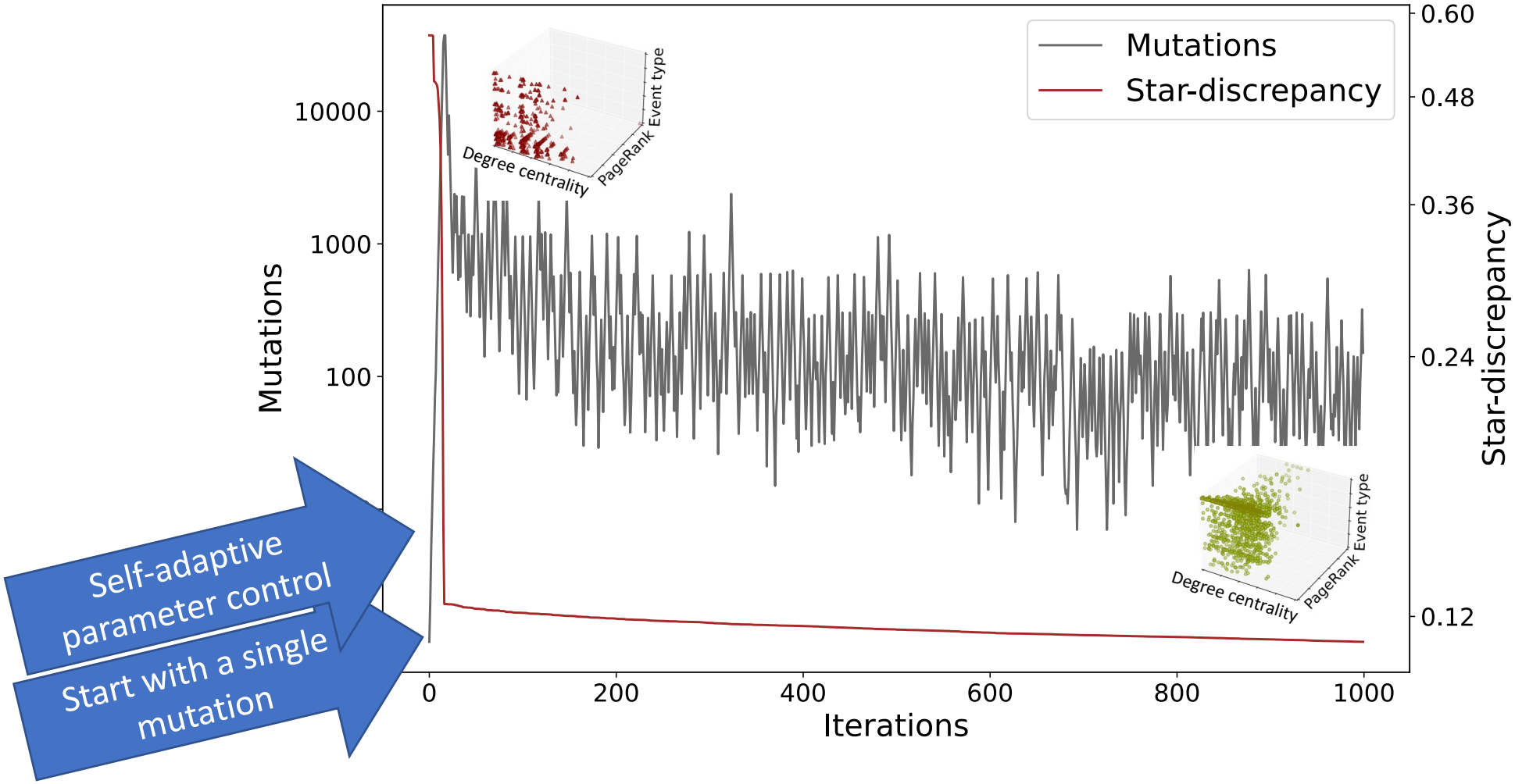
Stage 2/3: Evolutionary Diversification of Interaction



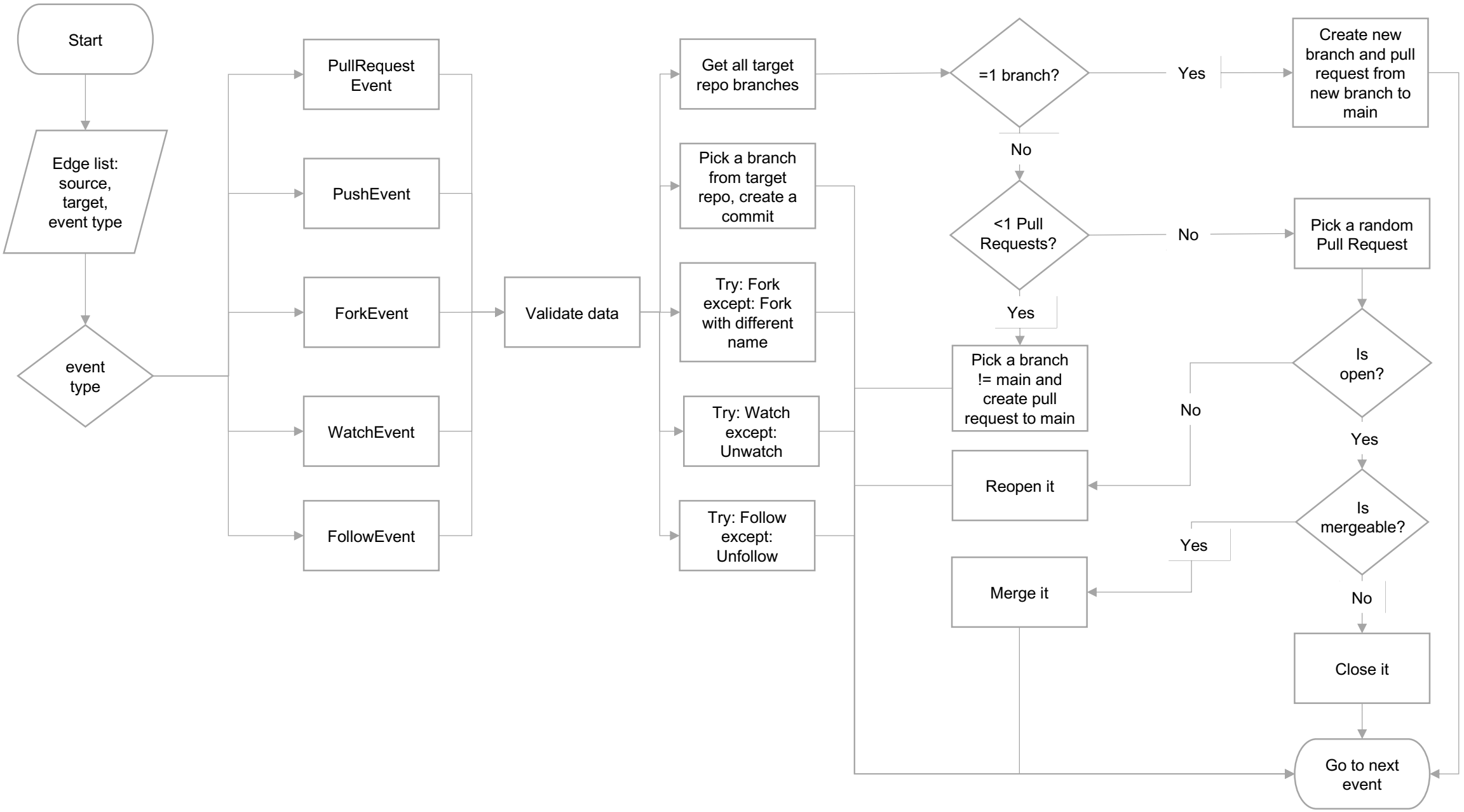
Stage 2/3: Evolutionary Diversification of Interaction



Stage 2/3: Evolutionary Diversification of Interaction

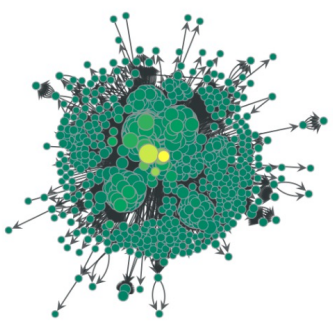


Stage 3/3: Execution and Eval.

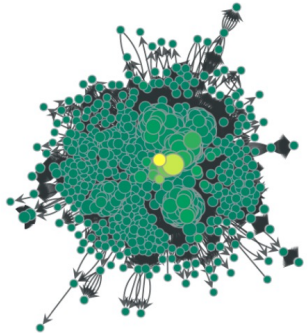


Stage 3/3: Execution and Evaluation

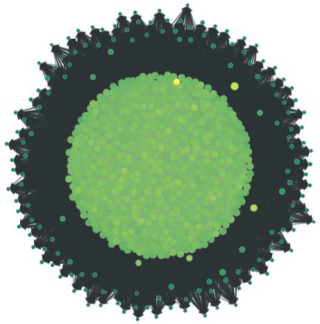
Before continuing… compare how and with what?



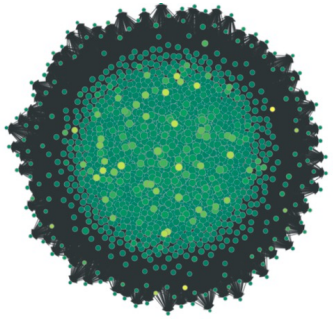
Original



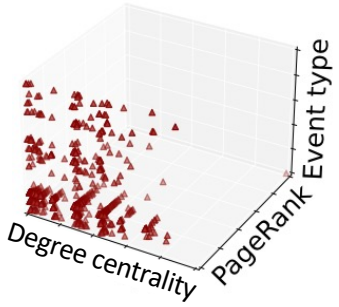
Simple



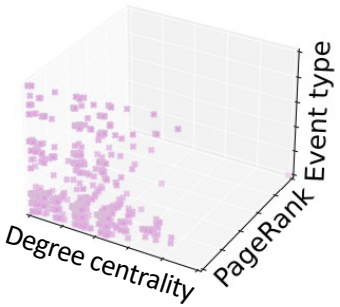
Random



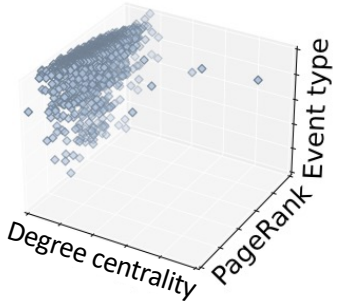
Evolved



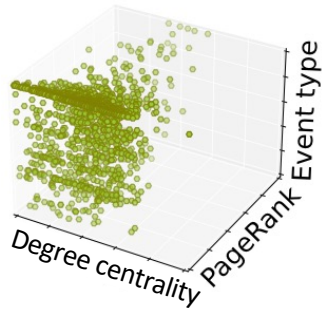
Original



Simple



Random

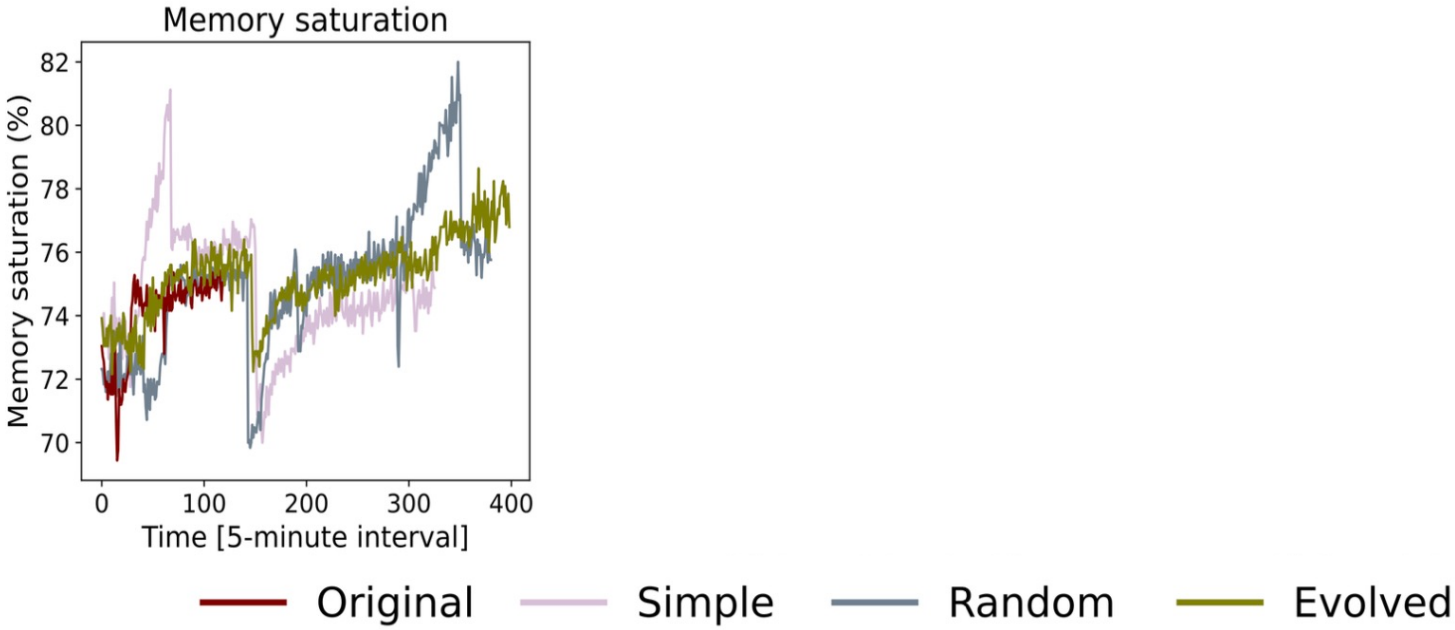


Evolved

Event type	Original
FollowEvent	195241
PushEvent	4234
WatchEvent	1206
PullRequestEvent	852
ForkEvent	450
Total	**201983**

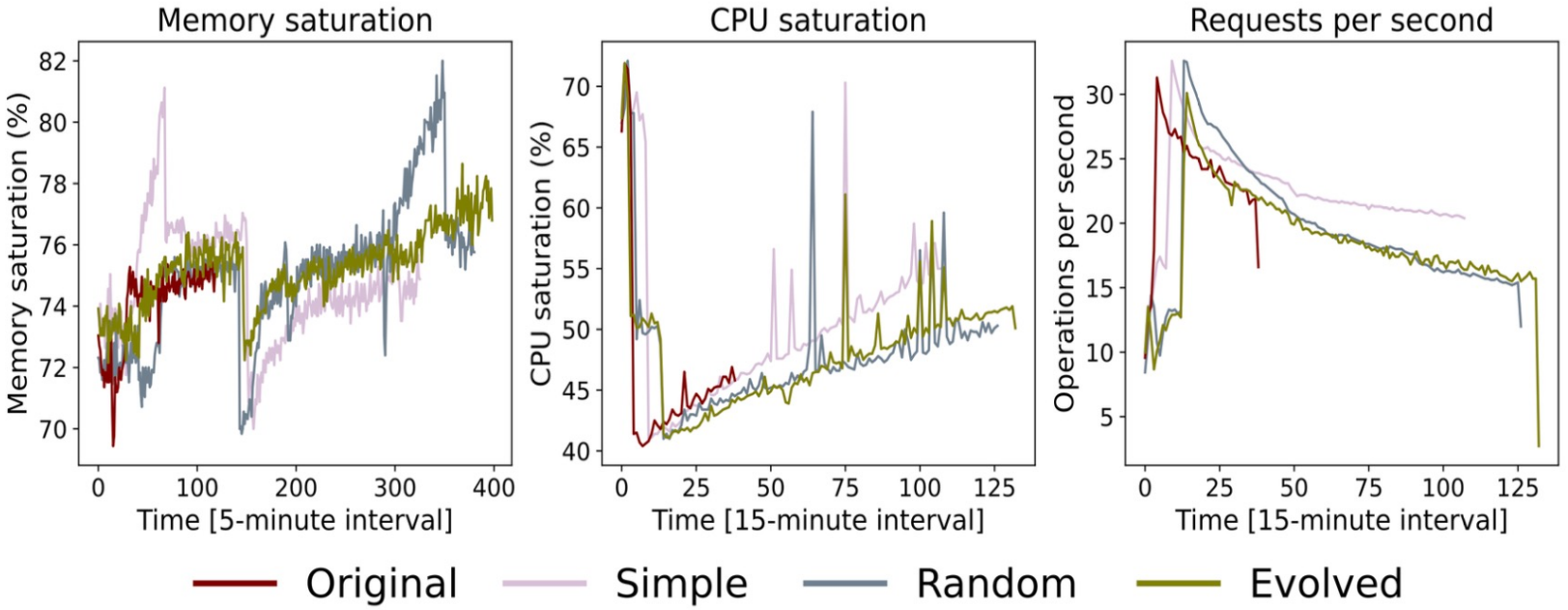
Observing effects

Effects on the system.

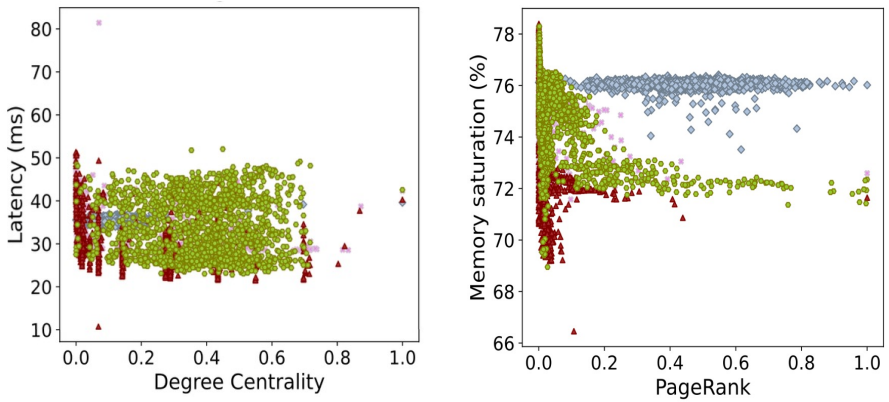


Observing effects

Effects on the system.



Correlation of effect and user characteristic:



Found one limitation: limit on number of users that can be followed.

Socialz – Summary

Key takeaways of this research:

- Social fuzz testing is a feasible approach, although the initial setup requires significant effort.
- Evolutionary diversity optimisation can generate community interactions that are significantly different from the original data or random data, potentially uncovering social bugs.
- Our testing also revealed a limitation that simple data replay could not.

Possible future work directions:

- Further characterisation and hybridisation of sub-communities.
- Exploration of additional community interactions and the related features.
- Integration of Socialz with traditional fuzz testing techniques that target code-level or system-level interactions.

We did it!

Conclusion

Lots of capable and user-friendly technology out there to optimise/visualise/learn/...

Lots of simple things you can try out at home – hill-climbers are your friends, even grid search can be your friend

@students: pay attention in your algorithms and maths courses ☺

http://acrocon.com/~wagner papers+slides online

markus.wagner@monash.edu

MONASH University