

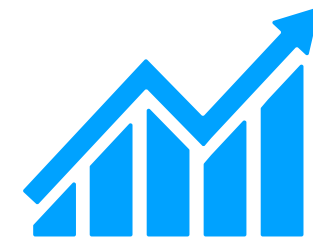
On Reducing Network Usage with Genetic Improvement

James Callan, William B. Langdon, Justyna Petke
University College London, UK



Increase in Network Use of Mobile Devices

Mobile traffic on cellular networks increased by 7 fold between 2016 and 2021 [33]

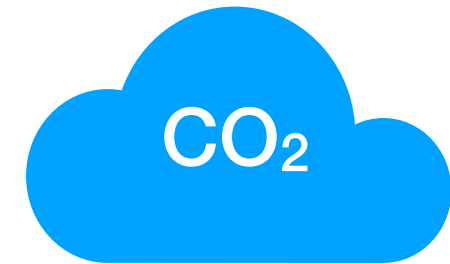


Excessive network usage is one of the most complained about issues of Android applications [21]

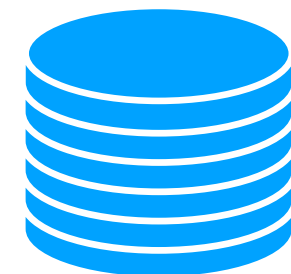


Energy from Network Use of Mobile Devices

Consumption of mobile phone communications could be responsible for up to 23% of greenhouse gas emissions by 2030 [5]



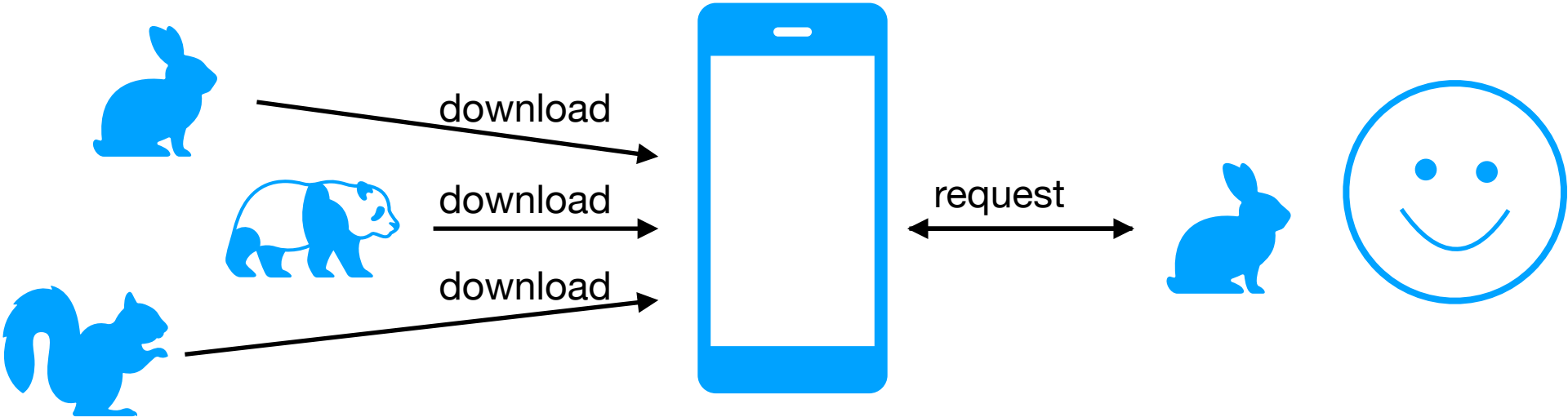
More expensive bills for user of mobile devices



Improving Network Use Via Prefetching

Before resources are needed

During Use



Unneeded resources downloaded :(

Improving Network Use Via Code Changes

http request

http request

http request

http request

http request



http request

Li et al. [23]

Reducing Network Use with GI

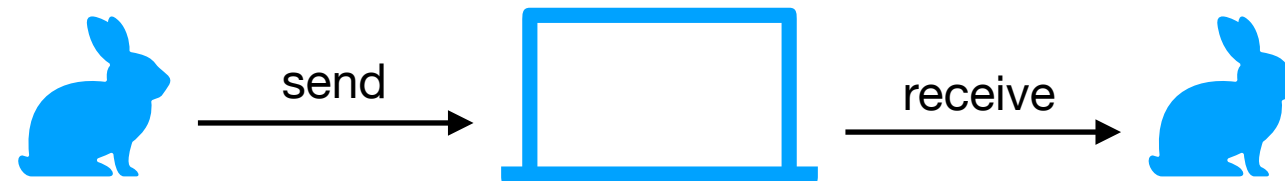
Previous multi-objective GI work by Callan and Petke [16] unsuccessful

Multi-Objective Improvement of Android Applications

James, Callan^{1*} and Justyna Petke^{1*}

^{1*}Computer Science Department, University College London, Gower Street, London, WC1E 6BT, Greater London, United Kingdom.

New Network Profiler




Automatically instruments 3 popular `http` Android libraries

Logs the size of data sent and received

Available at: <https://github.com/SOLAR-group/NetworkGI>



How do Android developers improve non-functional properties of software?

James Callan¹  · Oliver Krauss² · Justyna Petke¹ · Federica Sarro¹

Accepted: 11 February 2022 / Published online: 30 May 2022
© The Author(s) 2022

Abstract

Nowadays there is an increased pressure on mobile app developers to take non-functional properties into account. An app that is too slow or uses much bandwidth will decrease user satisfaction, and thus can lead to users simply abandoning the app. Although automated software improvement techniques exist for traditional software, these are not as prevalent in the mobile domain. Moreover, it is yet unknown if the same software changes would be as effective. With that in mind, we mined overall 100 Android repositories to find out how developers improve execution time, memory consumption, bandwidth usage and frame rate

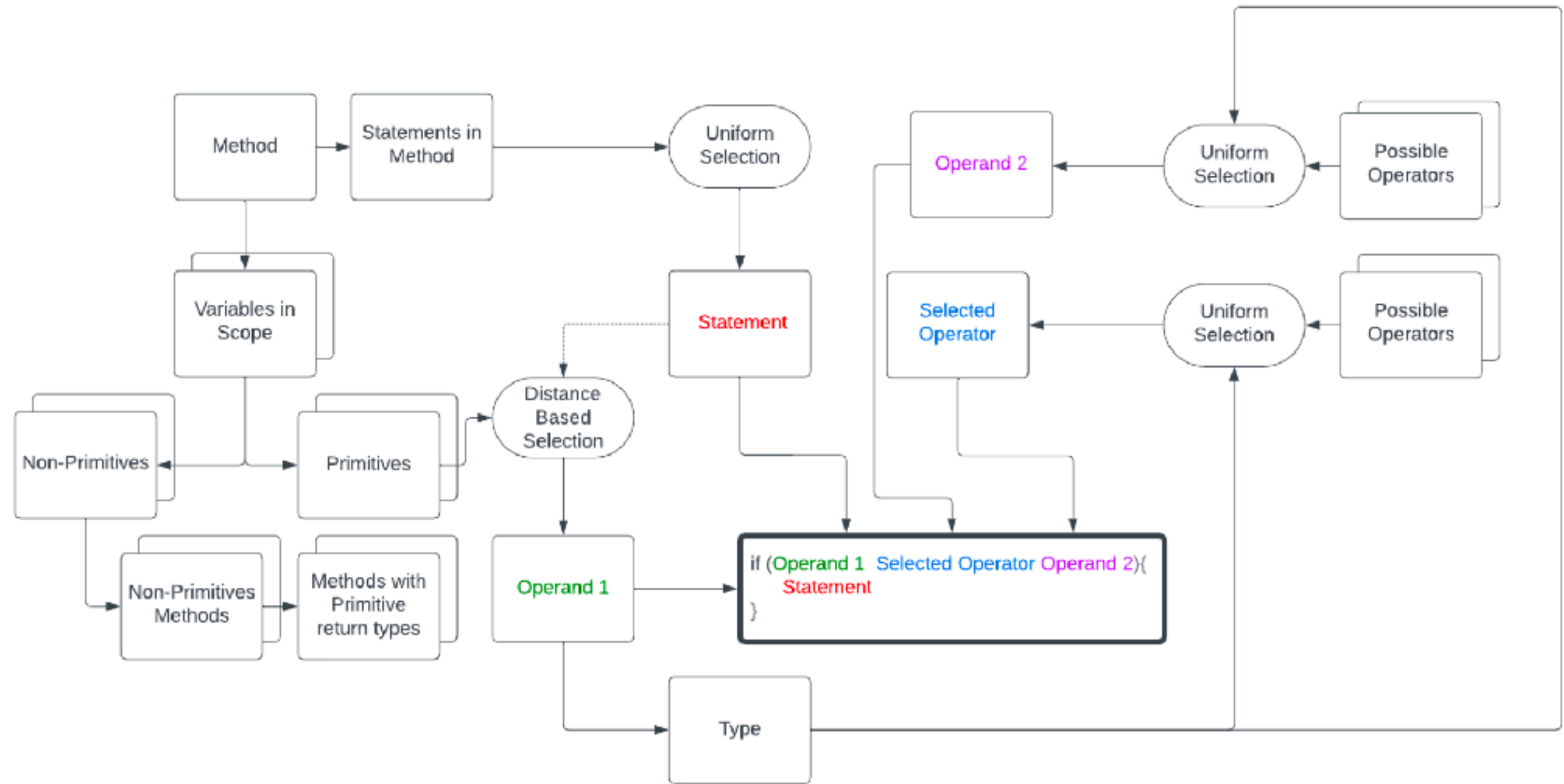
Removal of Unnecessary `http` Requests

```
...  
Asset asset = assets.get(0);  
Request request = new Request.Builder()  
    .url(asset.url)  
    .build();  
  
Response response = client.newCall(request)  
    .execute()  
...
```

```
...  
Asset asset = assets.get(0);  
Request request = new Request.Builder()  
    .url(asset.url)  
    .build();  
if (asset.isNeeded() == true) {  
    Response response = client.newCall(request)  
        .execute()  
}  
...
```

Added condition

Creating a conditional



Updated Set of Mutation Operators

Create a conditional

Delete a statement


Cache a variable value

Cache the result of a method call

Empirical Software Engineering (2022) 27: 113
<https://doi.org/10.1007/s10664-022-10137-2>



How do Android developers improve non-functional properties of software?

James Callan¹  · Oliver Krauss² · Justyna Petke¹ · Federica Sarro¹

Accepted: 11 February 2022 / Published online: 30 May 2022
© The Author(s) 2022

Abstract

Nowadays there is an increased pressure on mobile app developers to take non-functional properties into account. An app that is too slow or uses much bandwidth will decrease user satisfaction, and thus can lead to users simply abandoning the app. Although automated software improvement techniques exist for traditional software, these are not as prevalent in the mobile domain. Moreover, it is yet unknown if the same software changes would be as effective. With that in mind, we mined overall 100 Android repositories to find out how developers improve execution time, memory consumption, bandwidth usage and frame rate

Search Setup

Local Search (400 iterations)

Genetic Programming (40 individuals, 10 generations)

Each ran 20 times

Benchmarks

Covered by tests [28] + latest on FDroid

Use one of the 3 libraries

Issue: network-intensive methods uncovered by tests

Overall: 100+ Android apps -> 7 benchmarks

RQ1: Network Use Within Few Min Runs

Application	KLoC	Network usage (kB)	Most network-intensive method
Adaway	21.6	110.2	GitHubHostsSource.getLastUpdate
FDroid Client	88.5	237.9	FDroidApp.onCreate
GPS Logger	23.2	2.4	GoogleDriveJob.updateFileContents
Mi Mangu Nu	33.1	512.1	NineManga.getMangasFiltered
Materialistic	31.1	17.1	UserServicesClient.submit
F-Droid Build Status	7.1	1.5	FdroidClient.getRunning
Ooni Probe	32.7	147.6	MeasurementsManager.downloadReport

RQ2: Effectiveness of GIDroid for Network Use

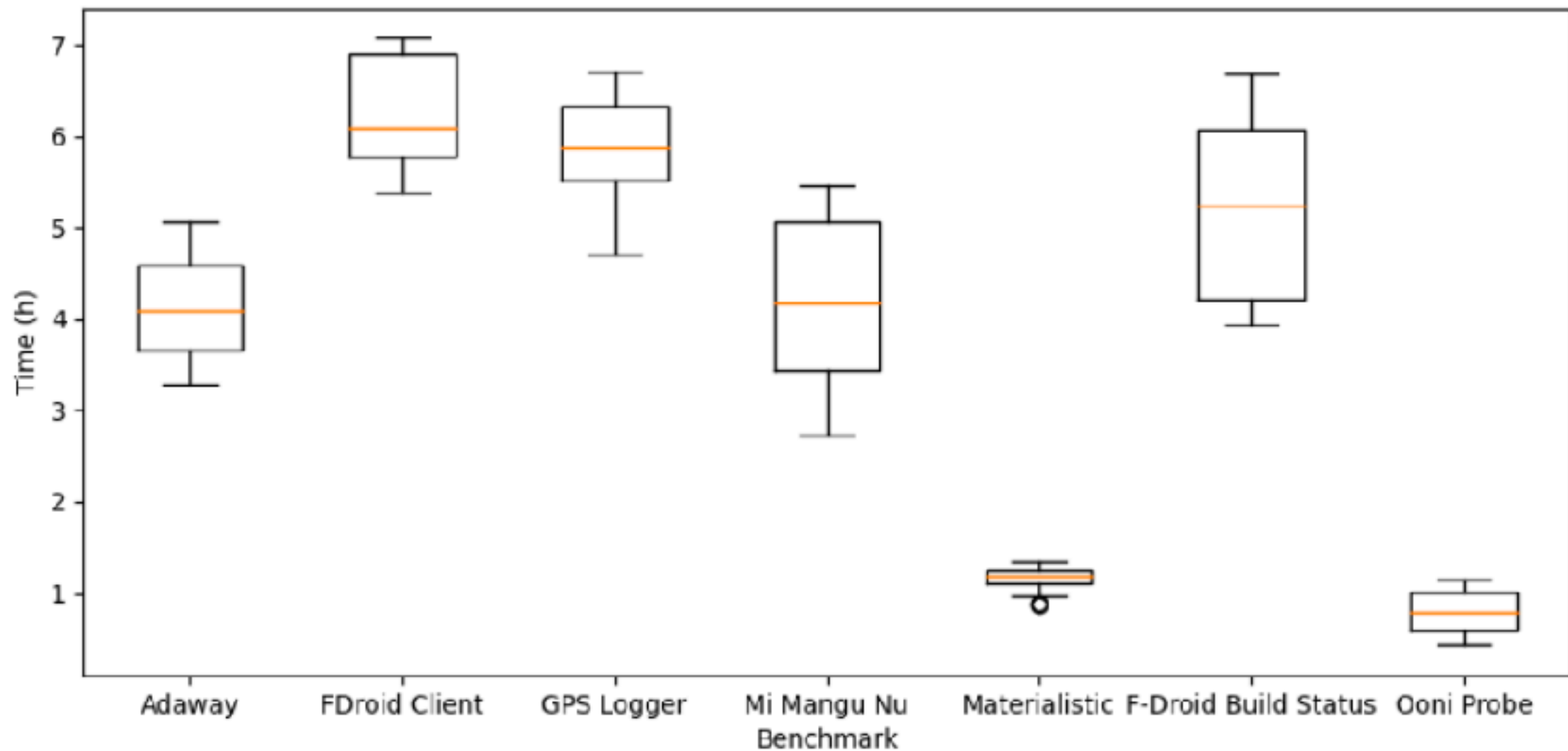
Still no improvements found :(

Reasoning:

search space too big (both for conditionals and overall)

RQ3: Efficiency of GIDroid for Network Use

Genetic Programming: median time: 4.3h (vs Local Search: 4.8h)



On Reducing Network Usage with GI

<https://github.com/SOLAR-group/NetworkGI>

