# Bi-partite graph experiments (version FF)

## The model

The model contains only "species" for both applications and platforms. There are no individuals.

Applications have a set of required services.

Platforms have a set of provided services.

An application has a set of links to platforms.

An application is alive if it is linked to a set of platform which provides all its required services (at least one linked platform provides each required service).

## The linking strategies

Application can be linked with any platform which provides at least one of its required services X. The meaning of a link is that this application could connect to this platform in order to be provided with X. In the bi-partite graph, there can be several strategies to create the links:

* Minimum set of links to satisfy all dependency. In practice that can correspond to a "non-adaptive" application. There are no unexploited links. This can be used a baseline.
* All potential links: This corresponds to an adaptive application with a complete knowledge of all platforms. It is the "perfect" adaptive application. It can be used as a "target" but is not realistic when systems scale up.
* Any intermediate strategy which might provide a more realistic situation: the application knows only about a sub-set of applications which are in its neighbourhood.

## The extinction sequence as a robustness metric

There is no re-linking during the extinction sequence. Platforms are killed in a random order. Applications are considered alive as long as they are linked to a set of platforms which provide all the services it requires. The extinction sequences need to be repeated with several random sequences of platforms in order to get a statistical result.

To evaluate the robustness with respect to different types of failures or attacked, the random extinction sequence can be replaced by other strategies. Ex: kill all platforms offering service X first or kill all platforms in the neighbourhood of Y, etc...

## EXPERIMENT #1

### Setup

* Same number of applications and platforms: 250
  + Rationale: To start somewhere.
* Each application required a random number of services chosen in a pool of 100 services.
  + Rationale: fewer that the number of applications.
* For each application, one platform is created to offer exactly the services required by the application. Applications are linked to only one platform: the one that was created for it and offers exactly what it needs.
  + Rationale: This is a baseline situation; applications are not adaptive and independent from each other's.

### Results

|  |  |
| --- | --- |
| Figure – Average Extinction Sequence  All extinction sequence give the same result: for each platform killed there is only one application connected to it which dies with it. | Figure 2 - Distribution of overall #service requests  This shows the number of times each service is picked to be required by an application. We are using a uniform distribution: all services have the same probabilities to be picked. It is verified on the data. |
| Figure - distribution of application size (#services)  This shows the distribution of application w.r.t the number of services they require. Here we are also using a uniform distribution: we have the same probability of having an application which requires between 1 and 99 services (not 0 or 100 because the implementation does not allow it). | Figure 4 - Distribution of #links (#apps)  This shows the number of links for each application. All 250 applications have 1 link as expected. |
|  |  |

### Conclusions