SumUp 23-10-2020

Mattia Milani*
*Dept. of Information Engineering and Computer Science, University of Trento, Italy
mattia.milani@studenti.unitn.it

I. Experiments Presentation

The goal of this document is, to sum up, and describe the experiments done up to now. All the experiments were done using the software in this repository and are fully replicable. How to run and analyze the experiments is out of the scope of this document.

The experiments are divided in two main categories:

- ***Single node evaluation***, in this group of experiments the goal is to analyze a single node evolution in the network;
- ***Network evaluation***, in this group of experiments the evaluation is done on the entirety of the network.

II. Goals

Like I specified in Section I all the experiments are divided into two categories that are distinguished by the size of the analysis. The simulation environment could be the same but the difference is in the analysis of the evolution.

In the first case, ***Single node evaluation***, the goal of the analyzer is to study a specific node and highlight the evolution of it. The output of the analysis could be the Finite State Machine (FSM) of the node and the signalling plot.

The signalling of a node represents all the possible outputs of a node. A single output signal represents in a single experiment the messages transmitted by the node, the result will be a mix of advertisements and withdraws in a string like "A1W1A4A6W6" This ouptuts signals, for each bunch of experiments, are collected in a CSV file with the appearance frequency of each output signal.

In the second case, ***Network evaluation***, we are looking for network results, evaluating the entire set of nodes and links. This is done by studying the number of messages transmitted and the convergence time.

Given T_{tx} as the time of the first transmission and T_{rx} as the time of the **last** reception the convergence time, CT, is given by the delta of those times: $CT = T_{rx} - T_{tx}$. The convergence is reached when the network becomes silent again.

III. Environments

Multiple environments have been used for the experiments. The main differences and properties of those environments are described in this section.

The first environment that I used is a *Fabrikant* environment with different Minimum Route Advertisement Interval (MRAI) settings. This name comes from the particular graph used, described in Section IV. The four types of MRAI used are:

- ***Fixed 30s***, MRAI is fixed for each link to 30 seconds;
- ***No MRAI***, MRAI is fixed for each link to 0.0 seconds;

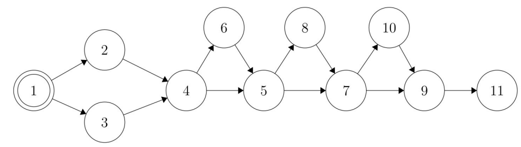


Figure 1: Fabrikant graph representation

- ***Ascendent***, MRAI will be doubled at each leach ($1-2-4-8-...$);
- ***Descendent***, Reverse of the ascendent case, MRAI will be divided by two at each leach.

The second environment used is a *clique* one, the graph is described in Section IV, all the parameters of the environment are described in Section V In all the clique experiments MRAI has been fixed in all the links. For each value of MRAI has been done 10 different experiments and MRAI goes from 0 in the first group of experiments to 60 in the last group. For a total of 610 runs in this topology.

The last one used is an *Internet Like* environment. The graph is described in Section IV, all the parameters are described in Section V. The main goal of this environment was to emulate, roughly, a real environment. The main difference between different experiments on this environment was the type of MRAI applyed:

- ***Random MRAI***, for each link the MRAI value will be chosen with a uniform distribution between $[0.0, MRAI_{limit}]$, the network must respect a defined $MRAI_{mean}$ value
- ***Fixed MRAI***, for each link the MRAI will be equal to $MRAI_{mean}$
- ***DPC***, for each link the MRAI will be setted according to the centrality of the node, process described in Section VI

In this environment has been run multiple experiments for each MRAI type. For each experiment, a new graph would be computed, so in the *Random MRAI*, for the same $MRAI_{mean}$ value could exists multiple different graphs.

IV. Input Graphs

In total three different base graphs has been used to produce all the results in this document.

A. Fabrikant Graph

The Fabrikant graph replicate what is described in the first figure in [1]. for simplicity, an example is reported here in Figure 1

Node 1 represent the only source of traffic, node 4 will prefer to reach the destination through node 2, but the link is slower, triggering changes in the network as required by [1].

Taking this base frabrikant graph other 4 graphs has been deveolped, one for each MRAI strategy applyed.

B. Clique Graph

The clique graph used for the clique experiemnts is composed of 15 nodes plus one external node that is the source of a destination. Each node is connected to every other node in a mesh network. The only node that does not respect this rule is the destination source. it has only one link that is connected to the node of the mesh network number 0.

Relationships between nodes are of the servicer type, so each node has 14 clients to updated when it receives an update. This ensures that the information is shared in the entire network.

For this network has been generated one graph for each fixed MRAI used, so that at the end we had 60 different clique graphs with the correct timer value equal on each link.

C. Internet Like Graph

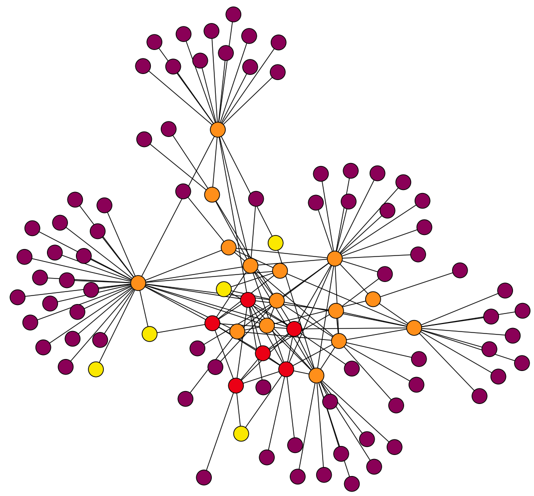
This network is composed of 100 nodes, it is not enough to emulate the Internet but, with enough computation time, the results should be comparable with bigger graphs. The graph has been produced following [2].

In the graph has been chosen only one node that shares a destination. The node has been chosen randomly in the set of clients nodes.

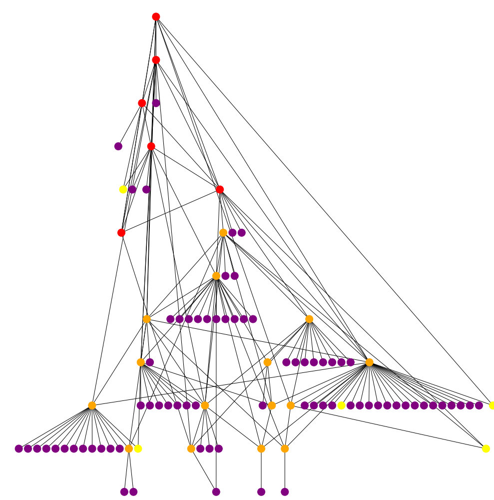
For each experiment, depending on the type of MRAI, a new graph file has been generated with different MRAIs values on the edges.

In total has been generated 10 000 internet like graphs.

The base graph is represented in Figure 2. The figure include two different layout of the graph, to show the hierarchical structure. Central nodes (red nodes) represent the tier one nodes, and them compose a little mesh network at the highest level of the network. Other types of nodes could be M, CP or C, the las category is the one representing the clients that can't do peering with other nodes and them are the magiority of the nodes. The random node that share a destination is chosen in the C set.



(a) Internet like graph with an "explosive" layout



(b) Internet like graph with a "hierarchical" layout

Figure 2: Internet like graph colored to show the hierarchicaly structure, 4 tipes of nodes, T (tier 1 mesh), M, CP, C (Customers, purple one)

V. Input arguments

In this section are described the inputs arguments used for the different environments.

A. Fabrikant arguments

- seeds: 20 different seeds;
- Signaling: "AWA", The input signal determines the messages that the source should send
- Implicit withdraw: active
- Withdraw distributions: 3 different withdraw uniform distributions $[5, 10]$, $[10, 15]$ and $[30, 45]$
- Reannouncement distributions: 3 different announcement uniform distributions $[5, 10]$, $[10, 15]$ and $[30, 45]$
- Processing time: constant with value $0.000\,01$
- Network delay: 3 uniform distributions in $[0.001, 1]$, $[0.5, 3]$ and $[2, 6]$

The number of possible different combinations of these values is 540, so for each different MRAI type has been done 540 experiments and in total 2160 expieriments in the fabrikant environment.

B. Clique arguments

- seeds: 10 different seeds;
- Signaling: "AW"
- Implicit withdraw: active
- Withdraw distributions: uniform distribution $[5, 10]$
- Reannouncement distributions: Ininfluent
- Processing time:i uniform distribution $[0.01, 1]$
- Network delay: uniform distribution in $[0.012, 0.1]$

This environment attempt to replicate what has been presented in [3]

In total this environment would run 10 different permutations because the only element that can differ is the input seed. But has been done in a total of 61 experiments changing the MRAI value between 0 and 60, so in total, we had 610 runs, 10 for each MRAI value. It wouldn't have much sense, in my opinion, to run more than one simulation batch per MRAI value, because the repetition of the seed with no difference in any other parameter would have produced the same result. For comparison purposes the same environemnt has been run even with the implicit withdraw option deactivated.

C. Internet like arguments

- seeds: 10 different seeds;
- Signaling: "A"
- Implicit withdraw: active
- Withdraw distributions: Ininfluent
- Reannouncement distributions: Ininfluent
- Processing time: constant with value $0.000\,01$
- Network delay: uniform distribution in $[0.012, 3]$

The environment has been used for random experiments and fixed mrai experiments.

Like in the clique experiments, in the case of the fixed MRAI, every link had the same timer value. But this time has been used also fractions of seconds to highlight the trend. There have been 121 experiments with MRAI in the ensemble $[0.0, 60]$. In the first fraction $[0.0, 5.0]$ has been used a step of 0.1 doing 51 experiments. The second fraction was $[5.5, 20]$ with a step of 0.5, doing in total 30 experiemnts. The last subset was $[21, 60]$ with a step of 1, doing in total 40 experiments. The final total is 121 experiments and for each of them has been done 10 runs, one for each possible seed of the environment.

For comparison purposes the same environemnt has been run even with the implicit withdraw option deactivated.

The second type of experiments with the internet like environemnt were run with the Destination Partial Centrality (DPC) MRAI strategy. This centrality metric si explained in Section VI, it's applyed on every edge of the network considering three phases in the graph. Then I applyed the same $MRAI_{mean}$ sequence that was described for the constant MRAI experiments, so in total we have 121 experiments.

For comparison purposes the same environemnt has been run even with the implicit withdraw option deactivated.

The last type of experiments with the internet like environment were run with random graphs. Before running a random experiment the $MRAI_{mean}$ were chosen randomly before the generation of the random graph. In total has been chosen 100 random $MRAI_{mean}$ uniformly distributed in the set $[0, 60]$ the limit 60 has been chosen arbitrarily being the double of the actual standard. 100 random graphs were generated for each $MRAI_{mean}$. Each link would obtain a random value in the set $[0, 240]$ and then all the values would be re-proportioned to respect the $MRAI_{mean}$ At the end for each random graph would be done 10 runs thanks to the 10 different seeds. The total number of this particular configuration is $100 * 100 * 10$ equal to $100\,000$ single runs.

VI. Destination Partial Centrality

In this section I will define and explain how was applyed the DPC in the experiments. What follow has been taken from [4].

The intuition at the base of centrality-based MRAI tuning is that we would like MRAI to increase in the initial phase, close to Tier one nodes set (from now on T_R set), and then, when the core of routers around T_R stabilized, it should start decreasing in order to quickly propagate the new stable situation to the rest of the Internet. To verify the validity of this intuition we set-up a strategy that exploits the previous knowledge of the network graph together with the concept of Destination Partial Centrality (DPC).

DPC is a variant of so-called load centrality which is defined in its general form as follows [5]: Consider a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and an algorithm to identify the (potentially multiple) minimum weight path(s) between any pair of vertices s, d. Let $\theta_{s,d}$ be a quantity of a generic commodity that is sent from vertex s to vertex d. We assume the commodity is always passed to the next hop following the minimum weight paths. In case of multiple next hops, the commodity is divided equally among them. We call $\theta_{s,d}(v)$ the amount of commodity forwarded by vertex v. The *load centrality* of v is then given by:

$$LC(v) = \sum_{s,d \in \mathcal{V}} \theta_{s,d}(v) \tag{1}$$

DPC adapts load centrality to represent the propagation of routes in an IP network. In DPC *load* represents the amount of networks that a BGP node exports. Not all nodes that run BGP generate *load*, but all nodes that forward traffic have a non-zero DPC centrality. We call $\mathcal{C} \subseteq \mathcal{V}$ the set of nodes that can be source and/or destination of traffic (they export at least one network) and N_s, N_d the number of networks that are exported by node s and d, respectively, then $\theta_{s,d} = \frac{N_s + N_d}{2}$. DPC $\Delta(v)$ of any vertex $v \in \mathcal{V}$ is defined as

$$\Delta(v) = \sum_{s,d \in \mathcal{C}} \theta_{s,d}(v) \tag{2}$$

With DPC we model the fact that some Internet routers export network addresses, and for this reason they generate changes in the network state, while other routers only forward traffic, but still their centrality can be larger than zero.

In a previous work we have shown that load centrality can be computed in a distributed way with minimal modifications to a Distance-Vector routing protocol. We also experimentally verified that computing DPC is possible with a custom BGP extension, and thus, it can be incrementally deployed on the Internet without requiring any global coordination [1]. Further theoretical details are outside the scope of this paper, but principles of centrality-based routing can be found in [6], [7].

Our proposal configures MRAI as a function of DPC with the following model: We assume the information contained in the `UPDATE` message propagates in the network in three phases, which identify three propagation graphs:

- ***Ascending phase graph*** $\mathcal{G}_\mathcal{A}(\mathcal{V}^{\mathcal{G}_\mathcal{A}}, \mathcal{E}^{\mathcal{G}_\mathcal{A}})$*:* made by the nodes updated without reaching tier one nodes;
- ***Tier one graph*** $\mathcal{G}_\mathcal{T}(\mathcal{V}^{\mathcal{G}_\mathcal{T}}, \mathcal{E}^{\mathcal{G}_\mathcal{T}})$*:* made by tier-1 nodes;

[1] A brief explanation on how to calculate in a distributed way the centrality can be found at: https://iof.disi.unitn.it/docs/DPCOnTopOfBGP.pdf

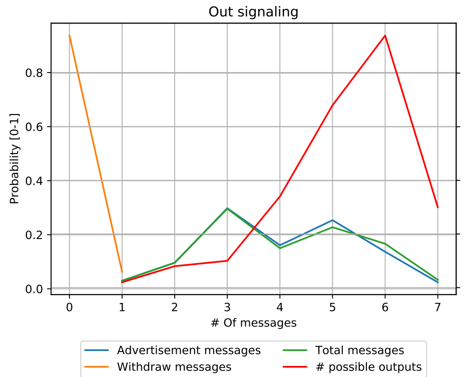


Figure 3: Fabrikant MRAI fixed 30 seconds, node 9 signalling output

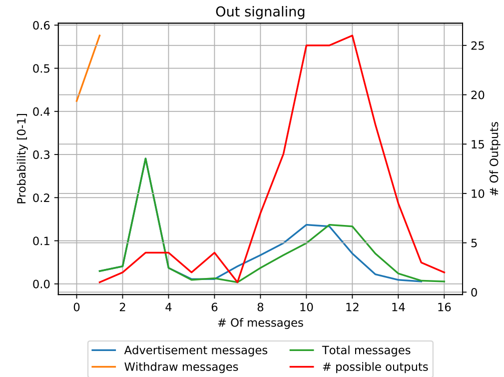


Figure 4: Fabrikant MRAI descendent, node 9 signalling output

- ***Descending graph phase*** $\mathcal{G}_{\mathcal{D}}(\mathcal{V}^{\mathcal{G}_{\mathcal{D}}}, \mathcal{E}^{\mathcal{G}_{\mathcal{D}}}) = \mathcal{G}(\mathcal{V}, \mathcal{E}) - \mathcal{G}_{\mathcal{A}}(\mathcal{V}^{\mathcal{G}_{\mathcal{A}}}, \mathcal{E}^{\mathcal{G}_{\mathcal{A}}}) - \mathcal{G}_{\mathcal{T}}(\mathcal{V}^{\mathcal{G}_{\mathcal{T}}}, \mathcal{E}^{\mathcal{G}_{\mathcal{T}}})$*:* the rest of the graph.

Considering a graph-wide maximum timer $T = 30\,\mathrm{s}$ and DPC $c_i \in [0, 1]$ for node i, DPC-based MRAI T_{ij} used by node i with neighbor j is set as follows:

$$T_{ij} = \begin{cases} \frac{T}{2}c_i & \forall i \in \mathcal{V}^{\mathcal{G}_{\mathcal{A}}} \\ \frac{T}{2} & \forall i \in \mathcal{V}^{\mathcal{G}_{\mathcal{T}}} \\ \frac{T(1-c_i)}{2} + \frac{T}{2} & \forall i \in \mathcal{V}^{\mathcal{G}_{\mathcal{D}}} \end{cases} \tag{3}$$

VII. Experiments Results

The first results that I would like to examine is the single node results from the fabrikant experiment. In Figures 3 and 4 are represented two signalling outputs of the node number 9.

The x axis represents the number of messages in the output signal. The first y axis, the one on the left, represents the probability to have a certain number of messages in the output sequence and should be used with "withdraw messages", "Advertisement messages" and "Total messages" lines. For example in Figure 3 we can see that there is a really high probability to have 0 withdraws in the output sequence, and we would never see more than 1 withdraw by the fact that the "withdraw line" doesn't go over that value of the x axis. In the same way, we can read the other lines, for example in the same figure is possible to see that the more probable output signal is composed by 3 messages, because is the highest point of the blue line. The second y axis, the one on the right, represents the number of **unique** states. This axis should be used with the line that represents the "possible outputs". Looking the Figure 3 is possible to see that for output signals with 6 messages we have more than 40 uniques output states.

Knowing that the output signalling is strictly dependent by the input that a node receives and the evaluation time of those messages we can already see the effects of an inconvenient MRAI setting. In Figure 4 there are a lot more output states. In those experiments we reach event output states of length 16 and the probability to have at least one withdraw is higher to the probability to don't have one. Knowing that the implicit withdraw system is active having one withdraw means that the node has no other possibilities to withdraw the network not knowing any other path.

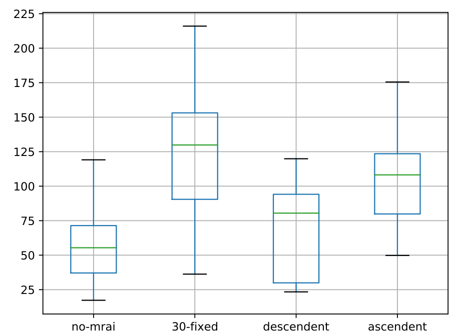


Figure 5: Fabrikant experiments convergence time comparison

Like we said before we used 4 different MRAIs strategies in the fabrikant environment, and them are compared in the Figures 5 and 6. The no mrai method is strictly dependent on delays and other events of the network and is possible to see that it has the smallest convergence time but with the higher number of messages necessary to reach the convergence. The 30 seconds strategy could be the slowest one because if something goes wrong is necessary to wait a long time to repair the damage, but it wouldn't require too many messages on the other side. The descendant method seems a good solution on the convergence time side, but on the other side, like is described in [1] it could easily lead to a lot of messages.

In Figures 7 and 8 are reported the general network results

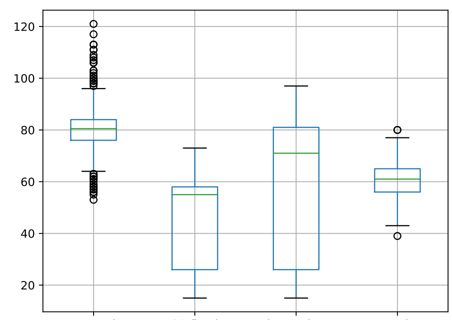


Figure 6: Fabrikant experiments number of messages comparison

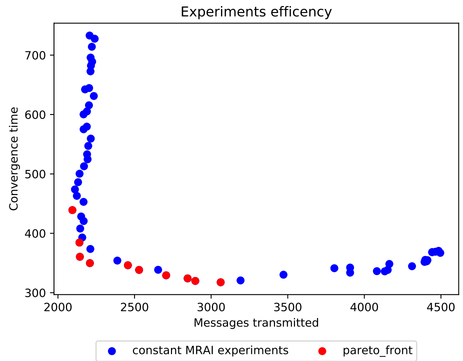
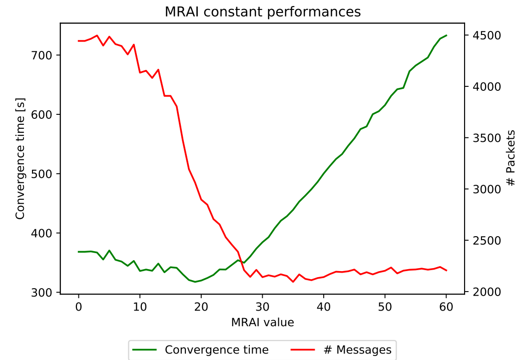
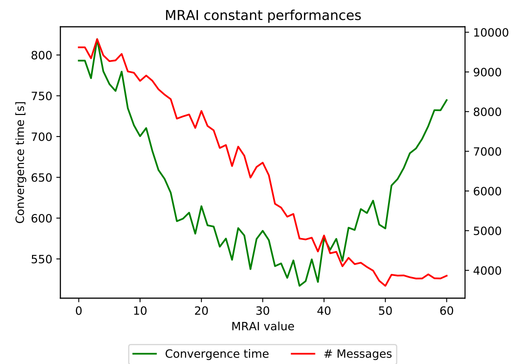


Figure 7: Pareto front in the clique environments



(a) Clique environment with Implicit Withdraw active



(b) Clique environment **without** Implicit Withdraw active

Figure 8: Evolution of the number of messages sent and the convergence time as MRAI grows in the clique environment

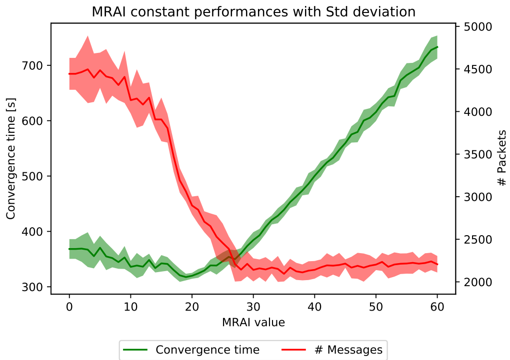
obtained in the clique environment. The goal of this study is to see in a general way how MRAI could influence even a small network as this clique of 15 nodes.

In Figure 7 every point of the plot is the mean of the 10 runs executed with a fixed MRAI. On the x axis is represented the number of messages correlated with the convergence time on the y axis. The red points are the Pareto front of the set of all points. We can see from the plot that a lot of experiments has a mean of messages sent around 2000 independently from the MRAI so we can guess that after a threshold of MRAI the number of messages stabilizes around that value. On the other side, before this threshold we can guess there is a lot of variance in the number of messages but the mean convergence time is similar. These guesses are confirmed by the Figure 8a where we can clearly see those trends. The two y axis are used to represent the trend of the convergence time and the number of messages transmitted in relation of the MRAI value. The two Figures 8a and 8b refers to two different situations in the same clique graph. In the first figure the implicit withdraw is active, in the second has been deactivated, the difference is evident, the number of messages and the convergence time are higher without the implicit withdraw active, and also take a longer time to reach a stable point in Figure 8b. This confirm what has been presented in [8].

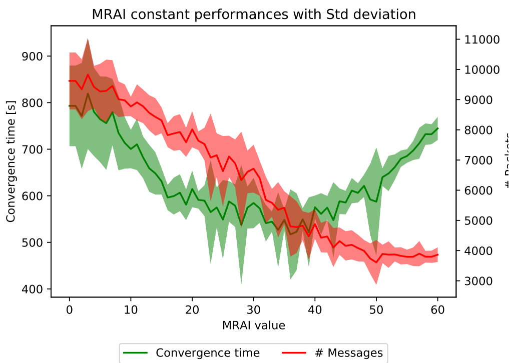
On the same environemnt I have produced also two more plots that are presented in Figure 9, the plots represent the same trend of Fig. 8 but this time there is a range around the lines that represent the standard deviation of every experiment (each experiment point is the mean of 10 runs). In Figure 9a is possible to see that with a small MRAI the standard deviation is higher respect to the standard deviation after a certain threshold of MRAI. The standard deviation of both messages and convergence time seems to be constant after a certain threshold. In Figure 9b is not possible to see clearly the same trend in the standard deviation, but with an MRAI higher than 50 is possible to guess that the standard deviation is starting to be more constant. Figure 9b confirms another time that without the implicit withdraw the number of messages and the convergence time would be a lot more suceptible and unpredictable.

The next experiment that I would like to study uses the fixed MRAI strategy on the internet like environment. The two Figures 10 and 11 have the same structure of the clique results but this time we can see a lot fewer messages transmitted to reach the convergence. Also this time we can see in Figure 10 that a lot of experiments are concentrated in the range between 146 and 147 messages with a high variance on the convergence time. In fact, we can see from Figure 11 that the trend is similar to the one that we saw in the clique graph, but this time the "# Messages" line has a steep fall, it reaches the constant state in few seconds (the clique experiment in Figure 8 took more than 20 seconds). But we can say the same thing for the convergence time too. Tose results are similar to the one available in [3]. With an MRAI lower than 1 s the number of messages could have huge spikes because the network delay is more influent than the timer. In fact, after a certain MRAI threshold the network delay becomes ininfulent and the number of messages strictly depend on the order of events that can produce constructive messages or disruptive that will be corrected in the next MRAI cicle. And this is the reason for the constant growing of the convergence time, it will takes always a higher time to correct incorrect messages.

The trend showed Figures 11a and 11b is comparable, in Figure 11b we can see that it requires more messages to reach the convergence but not more time. And the message trend



(a) Clique environment with Implicit Withdraw active



(b) Clique environment **without** Implicit Withdraw active

Figure 9: Evolution of the number of messages sent and the convergence time as MRAI grows in the clique environment, the range around the line represent the standard deviation of the experiment

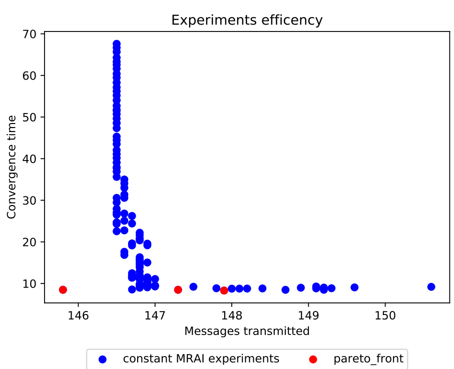
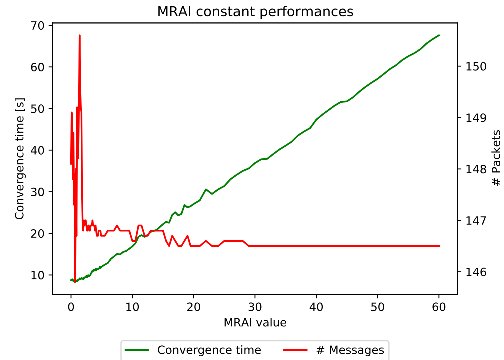


Figure 10: Pareto front in the Internet like constant MRAI environment

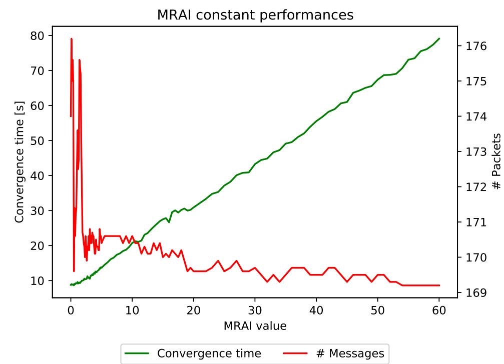
took more time to become stable.

Like before in Figure 12 is possible to compare the two techniques (Implicit Withdraw active, vs IW deactivated) with also the standard deviation of the experiments. Each point of the line is the mean of 10 runs, the standard deviation is calculated on those runs, those runs are just a subset of the infinite population of possible outputs.

The first thing that we can see in both figures is that the standard devition around the messages lines is constant for the majority of the time, execpt for a small MRAIs that produces



(a) Internet like constant MRAI environment with Implicit Withdraw active



(b) Internet like constant MRAI environment **without** Implicit Withdraw active

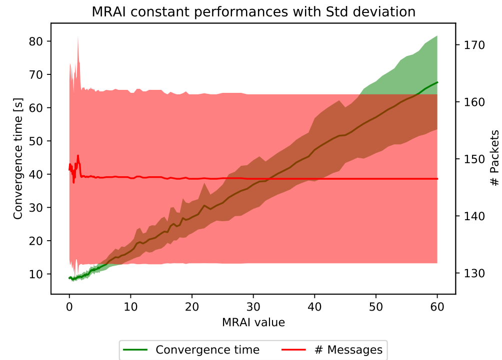
Figure 11: Evolution of the number of messages sent and the convergence time as MRAI grows in the **internet like** constant MRAI environment

huge spikes of variation. This effect is caused, like we said before, by the fact that after a certain MRAI threshold, the order of the events is the only factor that influence the number of messages. In all this experiments the only thing that changes was the MRAI value, so the order of events is the same (given by the seed of the RNG), and, given the ininfluence of MRAI, produces the same number of output messages in each run. It follows that that the standard deviation of each experiment is the same.

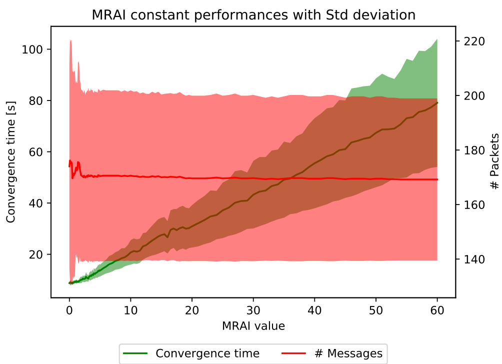
We have to remember that with the implicit withdraw feature are produced less messages by assumption, so we can't compare the number of messages or the standard variation of those thrends one another.

But, the convergence time is comparable. We can clearly see that the standard deviation of the time trend in Figure 12a is lower than the one in Figure 12b. I guess this difference is caused by the fact that huge storms of withdraws must be processed and eventually forwarded, causing more computation time and the forwarding of packets that don't give new knowledge at the neighbors. Packets that occupy a position in a FIFO link and with a delay.

I done the same experiments in the internet like environment



(a) Internet like constant MRAI environment with Implicit Withdraw active, standard deviation



(b) Internet like constant MRAI environment **without** Implicit Withdraw active, standard deviation

Figure 12: Evolution of the number of messages sent and the convergence time as MRAI grows in the **internet like** constant MRAI environment. The range around the lines represent the standard deviation of the experiment

with the DPC MRAI strategy. The number of experiments and the sequence of $MRAI_{mean}$ tested is the same as the constant MRAI.

Is possible to see in Figure 13 that the distribution is similar to the one in Fig. 10, but this time the spike of vertical values is around the messages value 149 with pratically no variation between one experiment and another. The horizontal spike this time is on the left of the vertical spike, and this is important because it represent that is more rare to go over that value of messages sent.

Like before we can guess that there will be some MRAI thresholds for the two spikes.

Once again the guess on the threshold is confirmed by Figure 14 plots where is possible to see that on low MRAI values we will have a huge variation on the number of messages distributed, but once the threshold is crossed the number of messages will be constant. And this time if we compare Figure 14a with Figure 11a we will see that with the DPC techinique this threshold is crossed way before. Is also noticeable that once the threshold is crossed there are no stability issues on the number of messages. exept for the

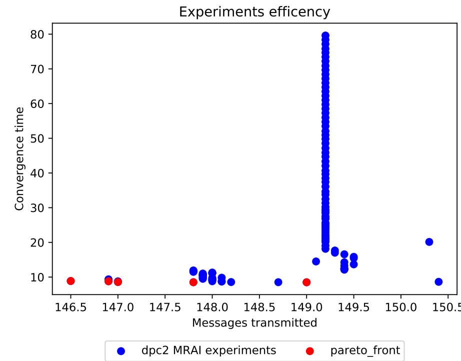
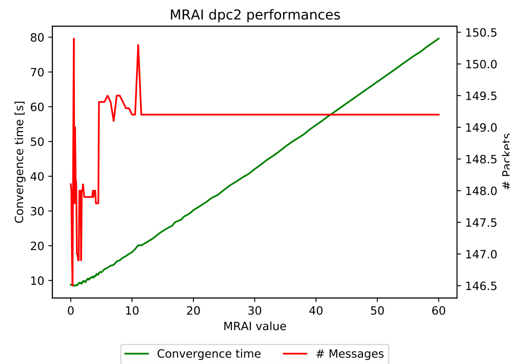
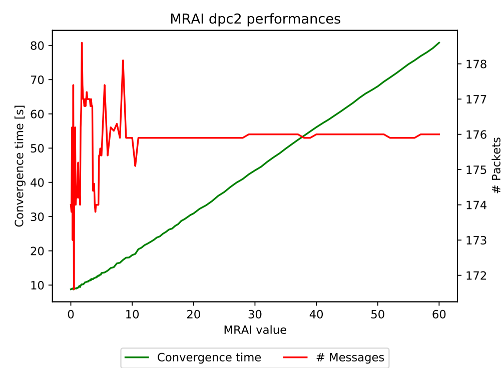


Figure 13: Pareto front in the Internet like DPC MRAI environment, IW active



(a) Internet like DPC MRAI environment with Implicit Withdraw active



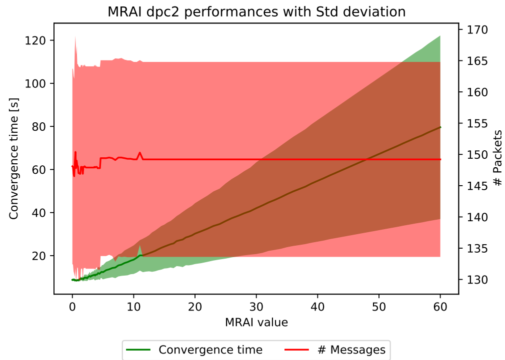
(b) Internet like DPC MRAI environment **without** Implicit Withdraw active

Figure 14: Evolution of the number of messages sent and the convergence time as MRAI grows in the **internet like** constant MRAI environment

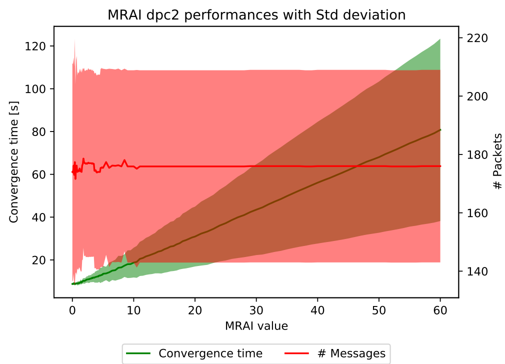
Figure 14b plot where we can see some small variation on the stabilized number of messages.

Like before in Figure 15 is possible to see the DPC MRAI strategy experiments on the internet like evironment with the standard deviation.

Like before the number of messages has a constant standard deviation after the threshold thanks to the event sequence.

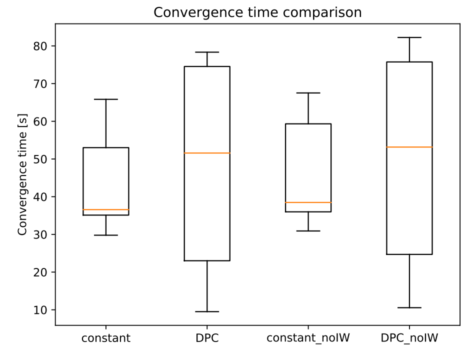


(a) Internet like DPC MRAI environment with Implicit Withdraw active, standard deviation

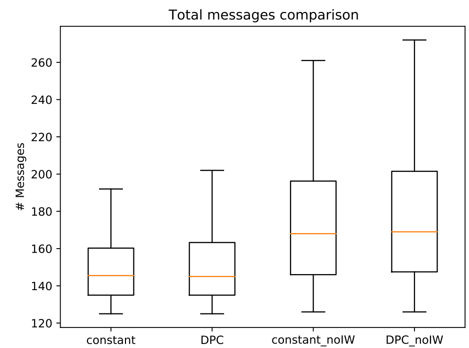


(b) Internet like DPC MRAI environment **without** Implicit Withdraw active, standard deviation

Figure 15: Evolution of the number of messages sent and the convergence time as MRAI grows in the **internet like** DPC MRAI environment. The range around the lines represent the standard deviation of the experiment



(a) Convergence time comparison between different environments



(b) Messages necessary to reach the convergence comparison

Figure 16: Comparison between the DPC MRAI strategy and the constant MRAI strategy, 100 experiments with the same conditions with $MRAI_{mean}$ equals to 30.0

The two plots respectively in Figures 15a and 15b are similar one another, execpt for the number of messages transmitted that it's not comparable. The big difference that we can see from the constant MRAI experiments is that the standard deviation around the time, in the DPC MRAI strategy, is bigger than the other. An educated guess could be that we can have a lower convergence time because MRAI is higher only where necessary and correct informations would travel faster around the network. On the opposite side bad informations could spread too, generating the opposit case with an high convergence time, caused by the fact that take more time to make every node converge.

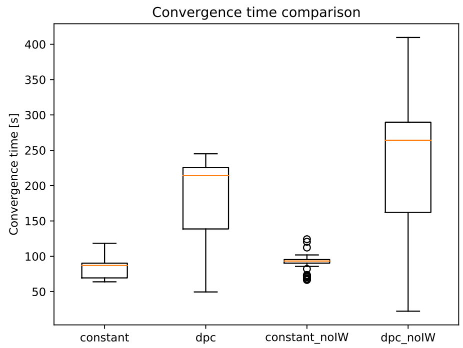
I then decide to take a look more closely to those MRAI strategies that we just saw. This time with $MRAI_{mean}$ fixed to 30 s for every MRAI strategy. I fixed the graph, using the internet like graph of the internet like environment. I increased the number of seeds that would be given to the random number generator of the des, so I was able to run 100 runs for each MRAI strategy. So I have done 100 runs for constant MRAI (with implicit withdraw and without) and 100 runs for the DPC technique (with and without implicit withdraw). The results are showed in Figure 16.

In Figure 16a are presented the boxplot to compare the convergence time of the experiments sets. The first thing that is possible to notice is that in the constant case the difference caused by the use of the implicit withdraw is marginal, just few seconds. On the other side, with the DPC the difference is huge. With the IW active, like we said before, we can reach a small convergence time thanks to positive information dissemination, but in case of disruptive messages this is can cause the opposite situation.

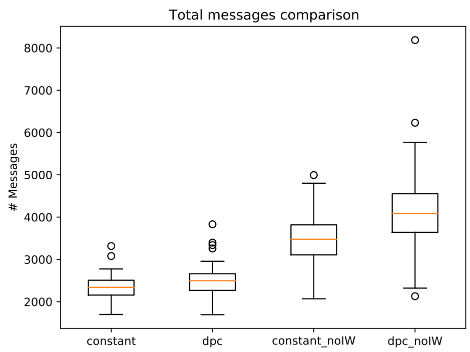
I have to say that I guess this situation is amplified by the fact that we are using a small graph not representative of the internet with a lot of nodes with MRAI setted to 0 sharing a lot of disruptive information without waiting enough.

So i decided to run the same experiments on an internet like topology of 1000 nodes. Always with $MRAI_{mean}$ of 30 and 100 different seeds. The results are presented in Figure 17.

Some more general results could be saw taking into consideration the random MRAI strategy on the internet like graph. The results are visible in Figure 18. The first thing that we can see is that this time there are not huge spikes for certain messages amount. The number of messages is more distributed between 145 and 152, with some particulary dense columns.



(a) Convergence time comparison between different environments



(b) Messages necessary to reach the convergence comparison

Figure 17: Comparison between the DPC MRAI strategy and the constant MRAI strategy, 100 experiments with the same conditions with $MRAI_{mean}$ equals to 30.0, in an internet like graph with 1000 nodes

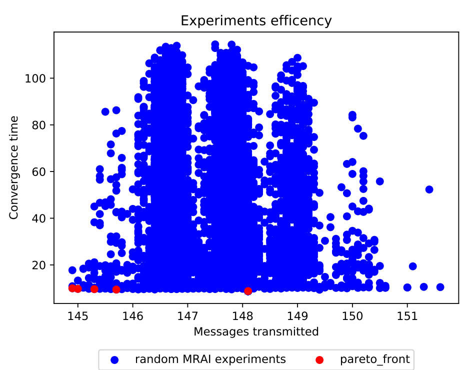


Figure 18: Pareto front in the Internet like random environment

Also, the convergence time is more distributed, thanks to the random MRAI distribution. We can guess that if a central node has a huge MRAI timer and it transmits incorrect information it would act as a bottleneck for the update with the correct information.

From the comparison of the three MRAI strategies we can see, in Figure 19, that the constant MRAI covers a really small part of the random strategy, and it has the tendency to produce a lot of messages for low $MRAI_{mean}$. On the opposite side

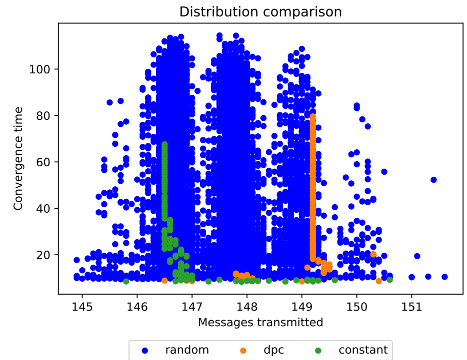


Figure 19: Comparison of the random strategy and the constant one in the internet like environment

the DPC strategy use more messages but the tendancy is to keep the number of messages under a certain threshold. And for small values of MRAI is also possible for the constant strategy to send more messages than the DPC strategy.

I would like to study more deeply the first part of Figure 18. In particular the experiments with convergence time lower than 20 and the number of messages under 146. We would see what particular MRAI settings those experiments has to produce those good outputs. The results are presented in Figure 20. In those plots we have on the x axis the nodes ordered by centrality. On the first y axis we can see the load centrality value normalized. The blue curve (that is equal on each plot, because the graph is one) represent the centrality of each nodes, thanks to the fact that nodes are ordered by centrality we see this line descending. Is also noticable that the magiority of the nodes have a centrality of zero. Each plot is divided in three sublplots that will help us to understand how MRAI is distributed in the graph. The red line in the first subplot represent the out MRAI mean of each node. The out MRAI of a node is the mean of the MRAI setted on the output links of the node. The second plot contains the input MRAI mean, that is the opposit of the output MRAI mean, so it's the mean of the MRAIs used by the neighbors with the node that we are analyzing. The last subplot contains the absolute difference of the previous two lines. Each point on those lines is the mean of all the graphs that produced results in the reange of our interest.

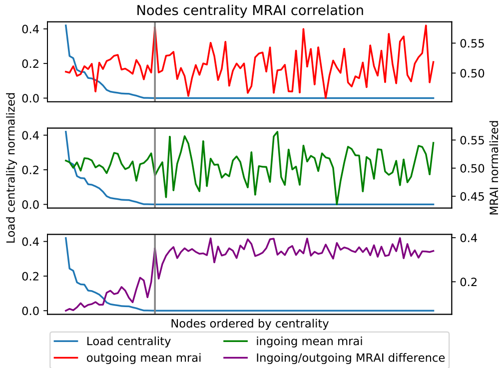
We can clearly see a correlation between the MRAI and the centrality. It would become even clearer using the second plot that contains also the sandard variation between all the means of the graphs in Figure 20b.

References

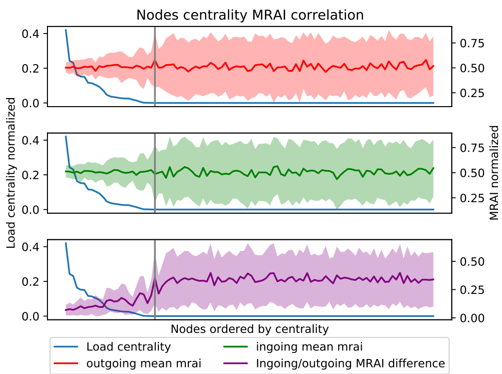
[1] A. Fabrikant, U. Syed, and J. Rexford, "There's something about MRAI: Timing diversity can exponentially worsen BGP convergence," in *30th IEEE Int. Conf. on Computer Communications (INFOCOM 2011)*, Shanghai, China, Apr. 2011.

[2] A. Elmokashfi, A. Kvalbein, and C. Dovrolis, "On the Scalability of BGP: The Role of Topology Growth," *IEEE Jou. on Selected Areas in Communications (JSAC)*, vol. 28, no. 8, pp. 1250–1261, 10 2010.

[3] T. G. Griffin and B. J. Premore, "An experimental analysis of bgp convergence time," in *Proceedings Ninth International Conference on Network Protocols. ICNP 2001.* IEEE, 2001, pp. 53–61.



(a) Study of the best results in the random experiments



(b) Study of the best result in the random experiments showing the standard deviation

Figure 20: Best random experiments, on the internet like topology, shows the correlation between centrality and the MRAI of the nodes, the gray wertical line represent the first node with centrality of 0

[4] M. Milani, M. Nesler, M. Segata, L. Baldesi, L. Maccari, and R. L. Cigno, "Improving bgp convergence with fed4fire+ experiments," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2020, pp. 816–823.
[5] U. Brandes, "On Variants of Shortest-Path Betweenness Centrality and their Generic Computation," *Social Networks*, vol. 30, no. 2, pp. 136–145, May 2008.
[6] L. Maccari and R. Lo Cigno, "Improving Routing Convergence With Centrality: Theory and Implementation of Pop-Routing," *IEEE/ACM Trans. on Networking*, vol. 26, no. 5, pp. 2216–2229, Oct. 2018.
[7] L. Maccari, L. Ghiro, A. Guerrieri, A. Montresor, and R. Lo Cigno, "On the Distributed Computation of Load Centrality and Its Application to DV Routing," in *37th IEEE Int. Conf. on Computer Communications (INFOCOM)*, Honolulu, HI, USA, Apr. 2018, pp. 2582–2590.
[8] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian, "Delayed internet routing convergence," *ACM SIGCOMM Computer Communication Review*, vol. 30, no. 4, pp. 175–187, 2000.