Content Distribution in VANETs using Network Coding: Comparison of Generation Selection Mechanisms

Alexander Afanasyev, SeungHoon Lee, Uichin Lee, and Mario Gerla
Computer Science Department
University of California, Los Angeles
{afanasev, shlee, uclee, gerla}@cs.ucla.edu

***Abstract*—In this paper we present the simulation-based evaluation of the generation selection mechanisms for multi-generation content distribution in Vehicular Ad-Hoc Networks (VANETs). The CodeTorrent system provides adaptation of peer-to-peer file sharing approach to the short-range nature of the vehicular networks by employing network coding mechanisms. However, due to the high computational requirement, it is impossible to perform single block network coding operation for big files. The trade-off of dividing files into generations solves the computational problem, but returns to the problem of the generation selection. A number of strategies can employed on the local nodes to provide appropriate global effect. The *local min*, *neighbor min* and *random* strategies optimize the global performance in terms of delay before whole downloading is finished. On other hand, the *local max*, *neighbor max* and *sequential* optimize for minimizing delay for individual generation downloading. In the paper we present the simulation results for these strategies and provide rationale of the observed effects.**

I. Introduction

The future deployment of the vehicular dedicated short range communication systems (DSRC) [1], opens the perspective of user demands for the variety of data transmission applications. The simplest example of these applications is software and multimedia downloads. Unfortunately, due to the short range nature of the DSRC and the high mobility of the potential service users, the conventional infrastructural approach can provide only limited informational service coverage. The solution to this problem is to make users to cooperate with each other to extend coverage beyond the limits of the infrastructural units. However, such cooperations creates a number of social, economical and technical problems, which should be solved before the real service can be deployed. In the paper we are focusing only on one aspect of the technical problem with the goal to minimize the service delay.

The intermittent connectivity of the vehicular networks limits usage of the most popular Internet peer-to-peer protocols. For example, BitTorrent [2] includes the dedicated central server for the peer discovery process, which cannot be directly implemented in the vehicle ad-hoc network (VANET) communication systems. The VANET by definition requires the pure peer-to-peer approach without relying on the support from the infrastructure. As a solution, Lee et al. [3] presented CarTorrent, where peer discovery was based on the multi-hop gossiping protocol. Similarly to the BitTorrent, the CarTorrent divides each file into a number of pieces and users employ some strategy to select piece for download. Integrated into the CarTorrent system AODV routing protocol [4] provides the ability to implement different piece selection strategies, such as “First Available”, “Rarest First” and “Rarest Closest”. All three strategies are based on the assumption that every piece is available at least some hops away. If the network is mostly partitioned with rare one-hop connectivity, like in the DSRC system, CarTorrent users would be unable reliably select the piece to download from one peer, which is useful for another one.

Another approach for content distribution is to employ the network coding [5], where instead of dissemination of individual file parts, their random linear combinations are distributed across the network. The reception of a predetermined number of linearly independent combinations (*the generation sizes*) ensures the possibility to recovery the original content. Previously, CodeTorrent system [6] was presented as an adaptation of the network coding technology to the BitTorrent-type content distribution within VANETs. Gaining the advantage of the piece selection freedom, it greatly suffers from processing overhead. The study [7] reveals that for 50 megabyte file the processing overhead is unacceptable and, in the terms of introduced delay for decoding the original file, is matching the actual downloading time. On the other hand, performing the network coding operations over the smaller individual file parts (*generations*) significantly decrease overhead to the acceptable level. The drawback of this approach is the return almost to the original CarTorrent problem for the piece selection strategy. The difference is that the CodeTorrent requires only a small number of generations that gives an acceptable overhead for the VANETs, and each generation can be divided into the any number of appropriate transmission size blocks without requiring any download selection strategy inside generation.

The global performance of the CodeTorrent system even with a small number of generations can be affected by the local decisions of the users. If each user would randomly pick available generations to download, then, from the global point of view, all generations should progress simultaneously. In the opposite case, if users would download all the blocks for one particular generation before trying to request something for another one, each generation will progress sequentially. There is a number of other strategies with different effects on the global performance, such as selecting generation with the least or the most downloaded blocks by node itself or among node neighbors. Our aim is to compare strategies through simulations and to determine with one gives the most optimal results for the individual and/or the global downloading progress.

The rest of the paper is organizes as follows. Section II gives the definitions of the implemented generation selection strategies for CodeTorrent file sharing system, Section III provides the excerpt from the simulations results and Section IV concludes the paper.

II. Generation Selection Strategies

In the multi-generation network coding systems like CodeTorrent it is very important to apply some generation selection strategy, which fulfills requirements for the particular application. One class of the applications may require minimal generation downloading time. For example, to provide uninterrupted multimedia broadcasting service the generation downloading time have to be less than playback time. Other class of applications, like file sharing service, expects the minimal overall downloading time for all generations. As we will show in Section III selection decision based on local or neighborhood generation downloading status predetermines global downloading progress.

Each node in the CodeTorrent system periodically broadcasts its downloading status for each generation. Depending on the gossiping protocol status packets may include (1) only binary indicators of generation availability or (2) numeric value for each generation representing individual downloading progress. In the latter case, the simple heuristic algorithm can be employed to filter nodes with small downloading progress, which improve chances to receive helpful (linear independent) coded block for particular generation.

In this section we will define 8 different strategies, which we were able to implement in our simulations. Two of the selection algorithms provide simple straightforward local-based strategies, another two make decision based on the local downloading status ranking. Next pair of algorithms make decision based on the neighbor aggregate download status ranking. Last two special algorithms (*Global Min* and *Global Max*) base their decision on the global downloading status, which we were able to obtain in our simulations. Using global strategies we are able to compare how local-based decisions approximate global decision.

A. Plain Strategies

1) Random strategy: A random selection decision among the all generations, which are not yet fully downloaded and available at the neighbors. Formally selected generation g can be defined as:

$$g = rand(\{i \mid L_i < S^{max} \text{ and } \exists k : N_i^k \geq S_i^{min}\})$$

where L_i – the local downloading status for generation i, S^{max} – the required number of the linear independent coded blocks to recovery the original content for each generation, S_i^{min} – the progress threshold for i^{th} generation after which node is considered to have useful data to download. For non-heuristic enabled simulations $S_i^{min} = 0$, for heuristics-enabled $S_i^{min} = L_i$, i.e., we are not considering node helpful if its downloading progress for generation i is less then our local progress for this generation. N_i^k – downloading status of k^{th} neighbor for generation i in the boolean (binary gossiping option) or integer format (extended gossiping option).

2) Sequential strategy: The generation with the least ordinal number g is always selected. If there are no neighbors, which can provide helpful blocks for this generation g, the available generation with the least ordinal number is selected instead. Formally it can be defined as:

$$g = min\{i \mid L_i < S^{max} \text{ and } \exists k : N_i^k \geq S_i^{min}\}$$

Figure 1 gives the key examples of the decision making for sequential strategy. No matter what is local or neighbor download status, the unfinished generation with the least ordinal number is selected. In special case, when this generation is not available in the neighborhood, the next least number is tried.

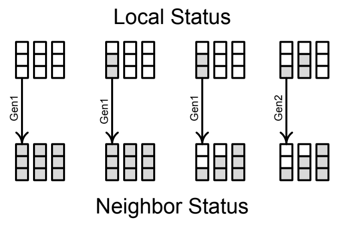

Fig. 1. Sequential strategy: the key cases

B. Local Rating Strategies

1) Local min strategy: The generation g is selected if the local downloading progress for this generation (L_g) equals to the minimum local progress value among all generations and at least one neighbor has at least the minimal progress (threshold S_g^{min}) for this generation:

$$L_g = \min_i(L_i) \text{ and } \exists k : N_g^k \geq S_g^{min} \Rightarrow g$$

Figure 2 shows the representative cases for the *local min* strategy. If the local download progress for all generations or generations with the locally minimal progress is the same, generation with the least number will be selected. In other cases, the minimal locally downloaded generation is selected provided some neighbor has passed the downloading threshold for this generation.

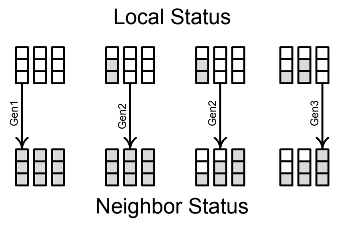

Fig. 2. Local min strategy: the key cases

2) Local max strategy: The most locally downloaded but not yet finished and available at least at one neighbor generation is chosen. That is, generation g is selected if the local downloading progress for this generation (L_g) equals to the maximum local progress value among all unfinished generations and at least one neighbor has at least the minimal progress (threshold S_g^{min}) for this generation:

$$L_g = \max_i(\{L_i \mid L_i < S^{max}\}) \text{ and }$$
$$\exists k : N_g^k \geq S_g^{min} \Rightarrow g$$

Figure 3 shows the representative cases for the *local max* strategy. If the local download progress for all generations or generations with locally maximal (but not yet finished) progress is the same, generation with the least number will be selected. In other cases, the maximal locally downloaded generation is selected provided some neighbor has passed the downloading threshold for this generation.

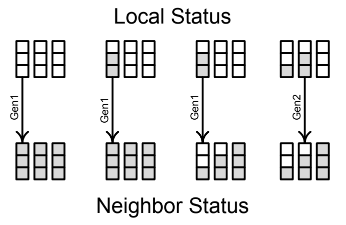

Fig. 3. Local max strategy: the key cases

C. Neighborhood Rating Strategies

1) Neighbor min strategy: The least downloaded by the neighboring nodes and locally unfinished generation is chosen. That is, generation g is selected if it has the minimal but positive aggregate download progress of the neighbors ($N_g = \sum_k N_g^k$) and at least one neighbor has at least the minimal progress (threshold S_g^{min}) for this generation:

$$N_g = \min\{N_g \mid L_g < S^{max} \text{ and } N_g > 0\} \text{ and }$$
$$\exists k : N_g^k \geq S_g^{min} \Rightarrow g$$

Figure 4 shows the representative cases for the *neighbor min* strategy. If the aggregate neighborhood download progress for all generations or generations with the minimal progress is the same, generation with the least number will be selected. In other cases, the minimal aggregately downloaded generation is selected provided some neighbor has passed the downloading threshold for this generation.

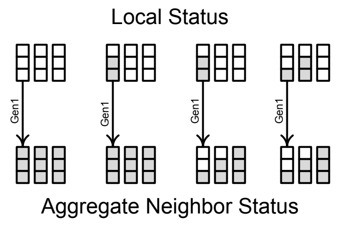

Fig. 4. Neighbor min strategy: the key cases

2) Neighbor max strategy: The most downloaded by the neighboring nodes and locally unfinished generation is chosen. That is, generation g is selected if it has the maximum aggregate download progress of the neighbors ($N_g = \sum_k N_g^k$) and at least one neighbor has at least the minimal progress (threshold S_g^{min}) for this generation:

$$N_g = \max \{N_g \mid L_g < S^{max} \text{ and } N_g > 0\} \text{ and }$$
$$\exists k : N_g^k \geq S_g^{min} \Rightarrow g$$

Figure 5 shows the representative cases for the *neighbor max* strategy. If the aggregate neighborhood download progress for all generations or generations with the maximum progress is the same, generation with the least ordinal number will be selected. In other cases, the maximum aggregately downloaded generation of the neighbors is selected provided some neighbor has passed the downloading threshold for this generation.

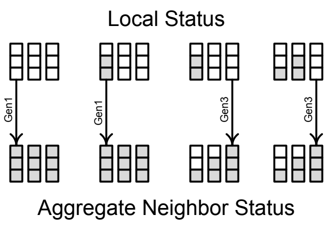

Fig. 5. Neighbor max strategy: the key cases

D. Global Rating Strategies

In the simulations we were able to perform generation selection strategies based on global rating. For this purpose, the special counter for each generation was implemented to individually track global downloading progress.

1) Global Min: The minimal globally downloaded, available at the neighbors and not locally finished generation is selected. That is, generation g is selected among the generations whose progress at neighbors is more than threshold, local progress less than maximum (S^{max}), and global progress for this generation (G_g) is minimal:

$$G_g = \min_i G_i \text{ and } L_g < S^{max} \text{ and }$$
$$\exists k : N_g^k > S_g^m in \Rightarrow g$$

Figure 6 shows the representative cases for the *global min* strategy. If the global download progress for all generations or generations with the minimal progress is the same, the generation with the least number will be selected. In other cases, the generation with globally minimal download progress is selected provided some neighbor has passed the downloading threshold for this generation.

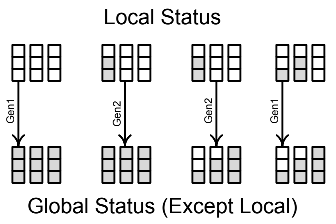

Fig. 6. Global min strategy: the key cases

2) Global Max: The maximal globally downloaded, available at the neighbors and not locally finished generation is selected. That is, generation g is selected among the generations whose progress at neighbors is more than threshold, local progress less than maximum (S^{max}), and global progress for this generation (G_g) is maximum:

$$G_g = \max_i G_i \text{ and } L_g < S^{max} \text{ and }$$
$$\exists k : N_g^k > S_g^m in \Rightarrow g$$

Figure 7 shows the representative cases for the *global max* strategy. If the global download progress for all generations or generations with the maximal progress is the same, the generation with the least number will be selected. In other cases, the generation with globally maximum download progress is selected provided some neighbor has passed the downloading threshold for this generation.

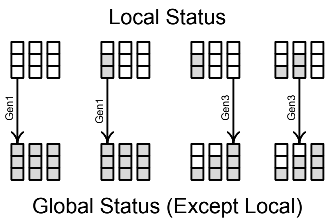

Fig. 7. Global max strategy: the key cases

III. Evaluation and Results

To answer the question which generation selection strategy can give the best results for overall downloading progress and each generation progress individually, we have performed a number of simulations using Qualnet network simulator[1]. For the physical and link layer transmission channel simulation IEEE 802.11b PHY/MAC model for was used. During the simulations, 200 nodes representing vehicle transceiver terminals, were moving

[1]Scalable Networks, http://www.scalable-networks.com

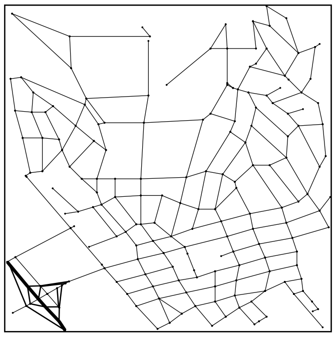

Fig. 8. Simulation area based on the 2400m x 2400m Westwood map

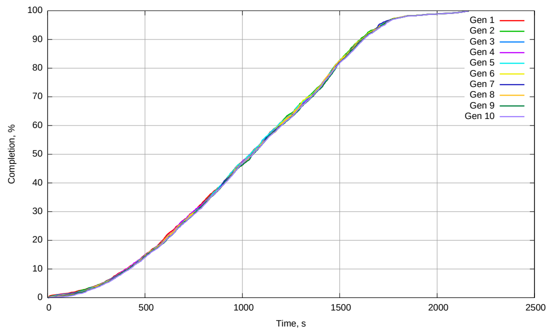

Fig. 10. Global downloading progress for each generation: Local min strategy

according to the Real Track model [8], which is based on Random Waypoint model by restricting node movements. This model gives more realistic results for urban areas and in our simulation we used $2400m \times 2400m$ *Westwood* street map area (Figure 8) as a restriction for node movements. The node speed varied in the range from 0 to 20 m/s. Three static nodes were representing the infrastructure and were providing the 50 megabyte file divided into 10 generations. Only 40 of the total 200 nodes were interested in the file downloading.

A. Parallel Generation Downloading Strategies

Figures 9, 10, 11 and 12 show the global downloading progress for each of the 10 generations of the 50 megabyte file, where *global min*, *local min*, *random* and *neighbor min* selection strategies were employed consequently. Randomness of the movements in the Real Track simulation model provides the effect of parallel generation downloading using these strategies. The *global min* (Figure 9) strategy represents the ideal global optimization function, where generations are requested in the way to minimize global imbalance between generations progresses.

In the *local min* strategy, as it can be seen in Figure 10, the local rating based decision pretty well approximates the global optimization function for parallel generation downloading. Moreover, the simple decision based on the *random strategy* (Figure 11) provide however more noisy, but still good approximation of the global optimization.

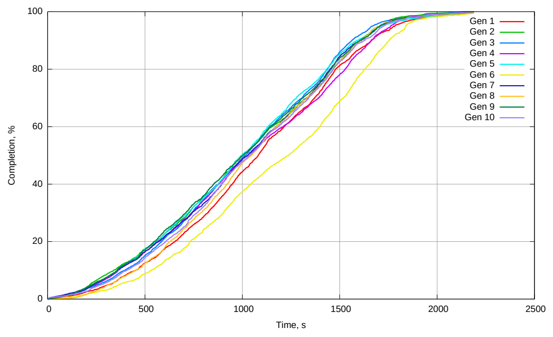

Fig. 11. Global downloading progress for each generation: Random strategy

The *neighbor min* strategy (Figure 12) gives more varied results for generation downloading progress. However, in the global view point, downloading times for each generation are arranged at the level of 2500 seconds, it is clearly visible, that in time period from 200s to 1000s download for the generation 1 was almost

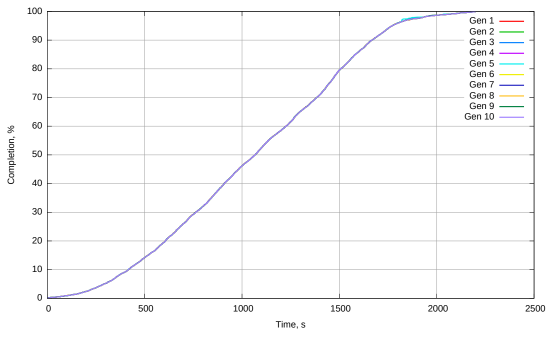

Fig. 9. Global downloading progress for each generation: Global min strategy

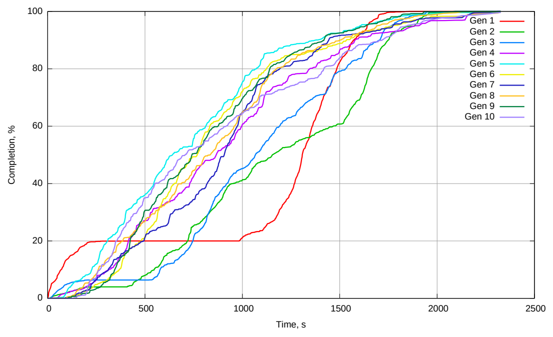

Fig. 12. Global downloading progress for each generation: Neighbor min strategy

stalled. The rationale for this behavior can be obtained by step-by-step examination of the generation selection strategy decisions. Initially, only access points (AP) have some data to offer. Due to the tie between "aggregate" generations downloading progress (in fact, every AP has all data for each generation), generation with the least ordinal number will be selected (i.e., the *Gen#1*). Thus, according to the strategy other generations can be selected for downloading only in two events: (a) if some node encounters at least the AP and one other node, which downloaded at least one block for the *Gen#1*; and (b) if some node have finished downloading from AP the *Gen#1*. The low probability of the former event describes the sharp starting downloading progress for the *Gen#1*. If a number of nodes, which have finished the *Gen#1* and started the *Gen#2* increased, strategy would force other nodes to select minimally downloaded generation, i.e., the *Gen#2*. The graphs for the *Gen#2* and *Gen#3* have similar trends (sharp start, stale period, etc), but due to the system randomization while progressing, strategy defined effects are faded. From the time moment 500 seconds every generation except the first start progressing in parallel, which fulfills the desired global optimization function.

Figure 13 shows the downloading progress for each selection strategy. The results were obtained by averaging results for 8 distinct simulation for each algorithm. Shown 95% confidence interval allows to argue, that initial downloading progress for *neighbor min* strategy is more robust than others, but around the 1200 seconds the graph slope start to decrease. It should be noted, that all these strategies has slow start-up and finishing phases (concave and convex graph forms consequently). The start-up behavior can be described with the fact, that parallel generation downloading in the begin of download progress forces to have minimal amount of coded blocks for each generation, which decreases chances for exchanging with helpful blocks. It is not so clear, why the global performance degrades near the end.

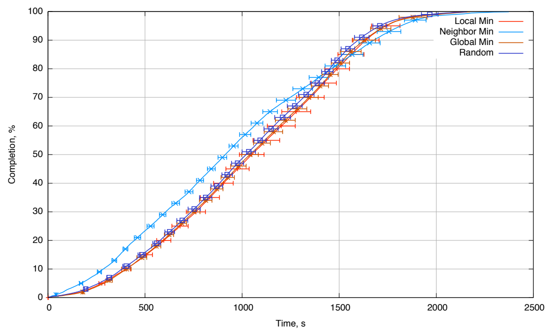

Fig. 13. Global downloading progress: parallel downloading of all generations

The histogram of download finishing times by each simulated CodeTorrent node presented in Figure 14. As it was expected, the *local min*, *global min* and *random* strategies gives more or less the same distribution of finishing times across the nodes, where the *random* strategy have a bit longer tail. Although the *neighbor min* strategy has the best starting download dynamics, it gives approximately 20% worse finishing times comparing to other strategies.

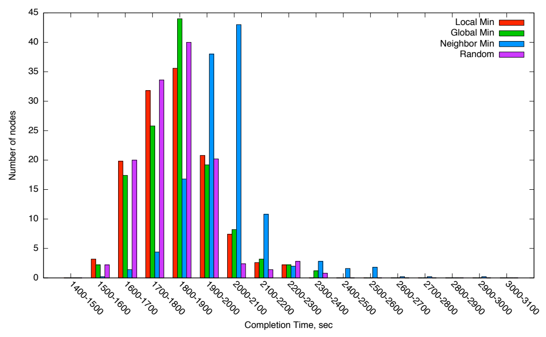

Fig. 14. Finishing time histogram: parallel downloading of all generations

B. Sequential Generation Downloading Strategies

Figures 15, 16, 17 and 18 show the global generations downloading progress for the *global max*, *local max*, *sequential* and *neighbor max* selection strategies consequently. These strategies give a virtually sequential downloading progress from the global point of view. The *global* strategy provides system-wide optimization for maximizing difference in generations download progress, which in most cases ensures downloading full one generation before starting second one. Due to the implementation specifics, in the *max* strategies we got the inverse order of the downloaded generation sequence.

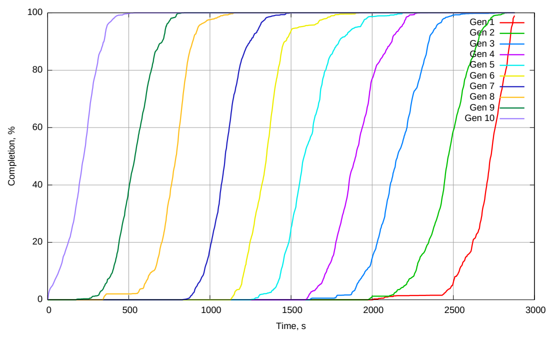

Fig. 15. Global downloading progress for each generation: Global max strategy

Unlike strategies from Section III-A, none of the strategies in current section provides ideal global function for true generation sequential downloading. If we draw vertical line at some time between 500s and 2500s, it is clearly visible that at least two generations are being downloaded in the system. Although this is helpful for overall downloading process, it limits startup performance (graph slope) for individual generation progress. Another effects is the absence of convex behavior at the end of the last generation downloading progress. That is, the individual generation progress benefits from the download exclusivity. This very important for broadcasting type applications.

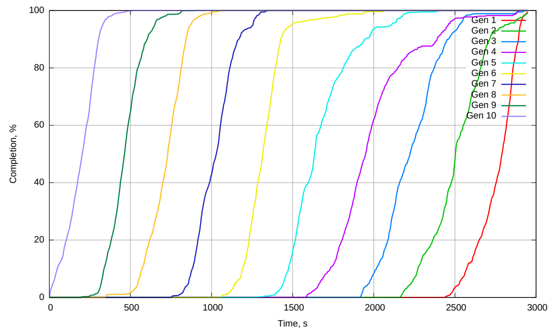

Fig. 16. Global downloading progress for each generation: Local max strategy

Enforcing the *sequential* strategy gives very similar system behavior to the *local max*. The only observed qualitative difference can be seen in the interval from 2500s to 3000s on Figure 16. The intersection of the *Gen#3* and *Gen#4* means that their individual progress becomes equal. Moreover, during the same time period, local rank for *Gen#3* were dominating *Gen#4*, resulting a bit faster download of the former. In the *sequential* strategy system is enforcing domination based on the ordinal generation numbers and such intersection situations are absolutely impossible.

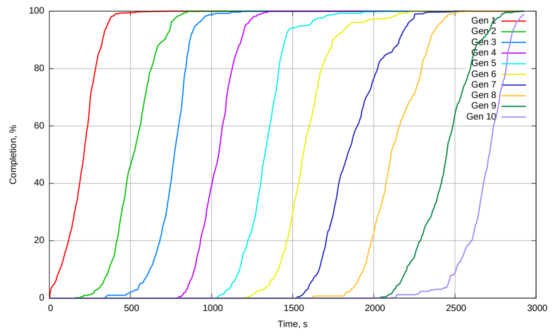

Fig. 17. Global downloading progress for each generation: Sequential downloading

Similarly to the *neighbor min* (Figure 12), the *max* strategy gives big variations in the downloading progresses. In Figure 18 we can see about 10 intersections, which means that randomness of the node encountering process can easy change global priority for generation selection. For some reason all generations have slow finishing dynamics.

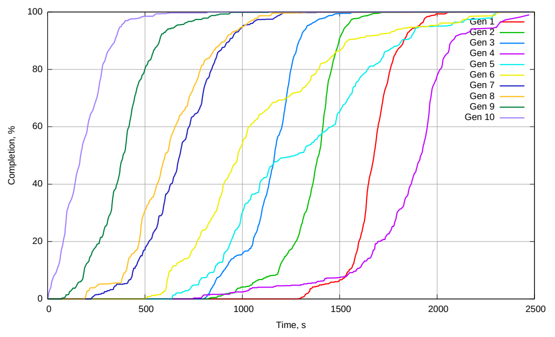

Fig. 18. Global downloading progress for each generation: Neighbor max strategy

Figure 19 shows the downloading progress for each selection strategy. The results were obtained by averaging 8 distinct simulation for each algorithm. Shown 95% confidence interval allows to argue, that initial downloading progress for *neighbor max* strategy is more robust than others, but around the 1800 seconds the graph slope start to decrease. It is interesting fact, that despite effects of "rare piece" at the start and the end of the each individual generation downloading progress (Figures 15–18), global progress for the *global min*, *local min* and *sequential* strategies is practically linear. This can be described by the fact, that when progress of one individual generation reaches "rare piece" phase, another generation have started its active downloading phase. Based on these observations, it can be claimed that parallel generation strategies described in Section III-A and the *neighbor max*, which virtually emulate parallel downloading are more effective than sequential ones.

Finishing times histogram (Figure 20) confirms outperformance of the *neighbor max* about 1.5 times comparing to other strategies. Other interesting fact is that the *global max* strategy has worse performance than

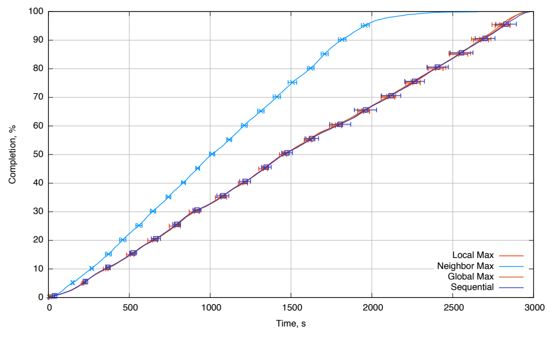

Fig. 19. Global downloading progress: sequential generation downloading

the *local max*, which can be described that local-aware strategy more relevant to particular node, than global one.

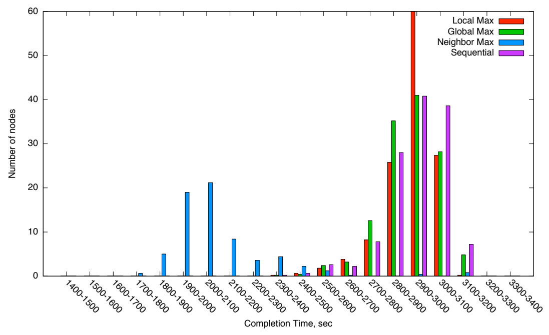

Fig. 20. Finishing time histogram: sequential generation downloading

C. Comparison of the Parallel and Sequential Strategies

Comparison of the global progress for the best strategies from the Section III-A and Section III-B is presented in Figure 21. At the beginning of the downloading process, both neighbor status ranking policies has similar performance. And as progress is heading to the end, their performance is starting to degrade. Though, the degradation level for the *neighbor max* strategy is higher.

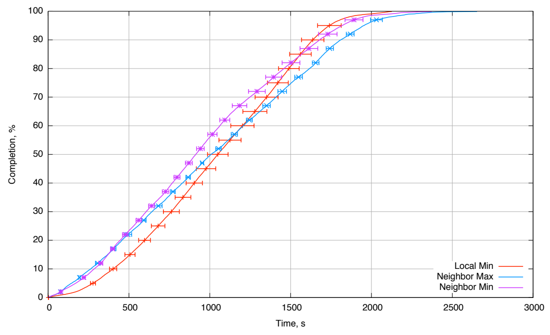

Fig. 21. Global downloading progress: comparison of the best from parallel and sequential strategies

Finishing histogram in Figure 22 shows, that despite the dominance of the neighbor strategies at the beginning, the *local min* strategy gives 20% better finishing times.

Theoretically, if it possible to detect the decrease of the global progress robustness (graph slope). The ideal performance improving combination is the *neighbor max* (from 0s to 400s), *neighbor min* (from 400s to 1100s), *local min* (from 1100s to 1800s) and *local max* to finish download process. We leave answer to the question how heterogeneous policy can improve system performance for the future research.

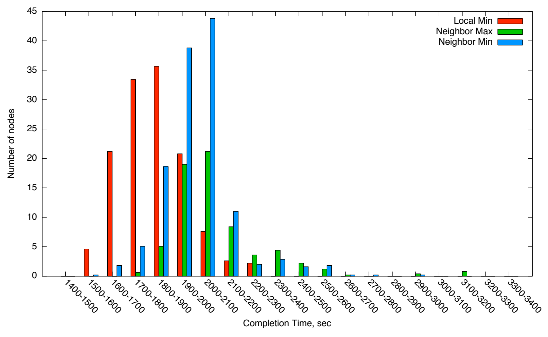

Fig. 22. Finishing time histogram: comparison of the best from parallel and sequential strategies

D. Influences of Gossiping Formats to the overall Downloading Progress

We have conducted a number of other experiments to compare effects of different gossiping protocols and effects of enabling heuristics in the generation selection process. Figure 23 present global downloading progress for the *neighbor min* and *neighbor max* policies. Two lines represent progress with enabled heuristics and two - with disabled. Employed heuristics filter out neighbors, who is unlikely to have helpful (i.e., the linearly independent) blocks. Formally, the heuristics can be defined as:

$$[\![\mathcal{N}']\!] = \{\mathcal{N}_k \mid \exists i : N_i^k \geq L_i\}$$

where $[\![\mathcal{N}']\!]$ – the set of useful neighbors, $\mathcal{N}_k$ – k^{th} neighbor, N_i^k – downloading progress of k^{th} neighbor for generation i, L_i – local downloading progress for generation i, k rolls over all neighbors and i – over all generations. That is, we eliminate all neighbors, whose progress for every generation is less than our local progress.

Simulation results in Figure 23 shows, that if we do not employ the heuristics, the download performance is greatly reduced. The best explanation is that it happens due to the increase of unhelpful blocks received from neighbors. Surprisingly, that the *neighbor min* strategy dominates the *neighbor max* if heuristics is enabled, and is dominated by the *neighbor max* without heuristics. Results presented in Sections III-A, III-B for neighbor-aware strategies was obtained with enabled helpfulness detection heuristics.

In addition, modification of the gossiping protocol can influence the global progress. For example, Figure 24 shows that change from distribution only the binary generations availability indicator (i.e., do we have something or not) to the sending detailed information about each generation progress (rank or completeness percentage)

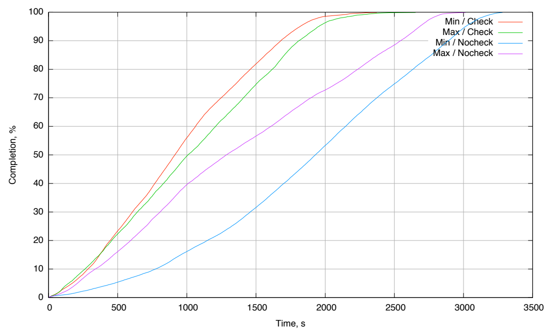

Fig. 23. Enabling (check) or disabling (nocheck) heuristics for neighbor selection

in the *sequential* strategy improve performance in the period from 1700s to 2500s. However, the finishing period is much worse in the latter case.

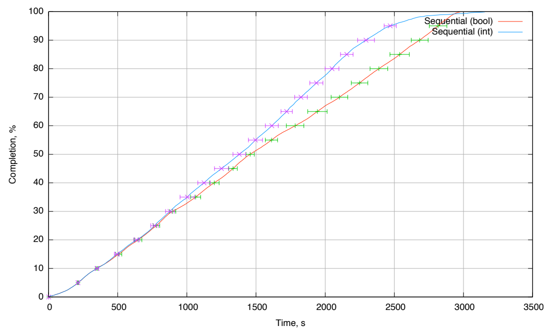

Fig. 24. Gossiping availability vector (bool) versus gossiping the available generation ranks

IV. Conclusions

We have defined and evaluated eight different strategies to select generation in the CodeTorrent file sharing system. However, two (*global min* and *global max*) strategies was implemented to provide reference point for ideal global optimization functions, they do not provide the optimal results in all cases. The global start-up performance is high if all nodes try to focus on particular generation, such as in the *sequential* or *local max* strategy. However, taking in account neighbor download status (the *neighbor min* and *neighbor max* strategies) can, and in fact does, help to further improve performance of the start-up phase. Simple *random* strategy gives comparable results to the *global min* and *local min* strategies for the individual generation downloading progress and statistically same results for the global download progress.

In the future work we will try to develop some heuristics to combine strategies to improve performance for file sharing operations. Theoretically, we can combine the best parts of graphs curves for global downloading progress (Figure 22). For example sequence *neighbor max*, *neighbor min*, *local min* and *local max* would result at least 20% faster than any other strategy.

References

[1] C. Cseh, “Architecture of the dedicated short-range communications (DSRC) protocol,” *48th IEEE Vehicular Technology Conference’98*, vol. 3, pp. 2095–2099, May 1998.

[2] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, “The bittorrent p2p file-sharing system: Measurements and analysis,” *Lecture notes in computer science*, vol. 3640, p. 205, 2005.

[3] K. Lee, S. Lee, R. Cheung, U. Lee, and M. Gerla, “First experience with CarTorrent in a real vehicular ad hoc network testbed,” *Mobile Networking for Vehicular Environments 2007*, pp. 109–114, 2007.

[4] C. Perkins, E. Belding-Royer, S. Das *et al.*, “RFC 3561 - Ad hoc on-demand distance vector (AODV) routing,” RFC, July 2003.

[5] P. Chou, Y. Wu, and K. Jain, “Practical network coding,” in *Proc of The Annual Allerton Conference on Communication Control and Computing*, vol. 41, no. 1. The University; 1998, 2003, pp. 40–49.

[6] U. Lee, J. Park, J. Yeh, G. Pau, and M. Gerla, “Code torrent: content distribution using network coding in vanet,” in *Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking*. ACM New York, NY, USA, 2006, pp. 1–5.

[7] S. Lee, U. Lee, K. Lee, and M. Gerla, “Content Distribution in VANETs using Network Coding: The Effect of Disk I/O and Processing O/H,” in *5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks 2008*, 2008, pp. 117–125.

[8] A. Nandan, S. Tewari, S. Das, M. Gerla, and L. Kleinrock, “Adtorrent: Delivering location cognizant advertisements to car networks,” in *Proc. Third IEEE/IFIP Annual Conference on Wireless On-demand Network Systems and Services (WONS06)*, 2006.