Optimal Mean Cost Replication in Desktop Grids

Ilya A. Chernov
IAMR KRC RAS, PetrSU
Petrozavodsk, Russia
IAChernov@yandex.ru

Abstract

In this paper we consider optimization problems for task scheduling in Desktop Grids; the used approaches has been weakly reported in literature so far despite their potential efficiency here demonstrated in simple estimations. We propose the mean-cost approach and show how it can be used for different purposes, including counter-sabotage and deadline safety, derive necessary inequalities for the costs, and show that task grouping can be efficient for optimizing extra losses but hardly can be used as a sort of replication.

1 Introduction

Mathematics of Desktop Grid task scheduling is quickly developing. It is enough to mention surveys [KMH17, ET12, DS14, XA10, CKBH06]. Much attention is paid to efficiency in terms of throughput, makespan, or the total time to complete a batch of tasks, and reliability defined as low risk of getting a wrong or invalid answer or not getting any answer at all, including sabotage-tolerate techniques. There are other points of interest, including availability of nodes, load balance or minimization of the peak load, and others, that are out of the scope of this work.

Obviously different criteria contradict or at least conflict with each other. Some effort has been paid to simultaneous improving two or more criteria. However, nobody (up to our knowledge) considered optimization of the average cost with all conflicting factors converted to the same unit (time, currency, etc). In [Che16, CN15] we studied such problems for the optimal replication. The only assumption about the computing system was that tasks are solved independently with a unit cost and the same risk of producing a wrong answer from the fixed set (due to malfunction, errors, malicious activity, etc). Each task is replicated until ν identical answers are obtained. If a wrong answer is accepted (this will sooner or later show up), come penalty F is added to the overall cost. This approach allows to choose the optimal replication level and also can be helpful for, e.g., choosing parameters of the computing algorithm. For example, choosing higher precision usually reduces the risk of an error, but needs more time; replication with a cheaper algorithm sometimes is more efficient, of course provided that calculations are independent.

Interesting results obtained here were quick growth of the critical penalties with respect to risk levels ($F \sim p^{-1}$), so that there is no need to know penalties precisely; low cost of overestimation of the optimal replication for rare (and thus valuable) results; and more careful check of less valuable answers.

Here we apply the approach to other cases, obtaining rather simple estimations that can be useful (we hope) for developing scheduling algorithms. Also we consider estimations for task grouping which can serve a number

In: E. Ivashko, A. Rumyantsev (eds.): Proceedings of the Third International Conference BOINC:FAST 2017, Petrozavodsk, Russia, August 28 - September 01, 2017, published at http://ceur-ws.org

of purposes: improving efficiency, checking, kind of "fractional" replication (only some tasks of a parcel can be double-checked). However, task grouping seems insufficiently studied in literature so far.

2 The mean-cost approach

2.1 Counter-sabotage scheduling

Assume that a wrong answer can be accepted due to malefactors and all that is known is the probability p to send a task to a malicious computing node. We use replication (ν copies) to reveal the malefactors: if at least two replicas produced different results, the task is re-solved and the saboteur is discriminated. In case a wrong answer has been accepted, some penalty F expressed in terms of the unit cost of a single processing of a task is paid. Then the average total cost is

$$E = \nu + p^\nu F.$$

This function has minimum either at $\nu = 1$ (no replication) or at some $\nu > 1$, because for very high ν it grows almost linearly to infinity. The necessary condition of the minimum is

$$p^\nu \ln \frac{1}{p} = \frac{1}{F}$$

which means that the optimal ν depends on logarithm of the penalty and the risk level. If p is small so that $p^2 \ll p$, we are able to derive simple tests. The replication $\nu + 1$ is better than ν if $p^\nu F > 1$. So, here again the critical penalties (that force the change of the replication level) are reciprocals of the risk level.

2.2 Deadlines

Consider a reliable computing system with equal nodes and pseudo-answers yes and no obtained if at least one replica/no replicas was solved before the deadline D expressed in terms of the unit solving cost for an answer. The probability of violating the deadline (i.e., the "no" answer) is p and if the "no" is "accepted", i.e., all replicas violated the deadline, some penalty is added to the overall cost.

Denote the number of replicas by ν. Probability of i answers got in time and $\nu - i$ not is $P = C_\nu^i q^i p^{\nu-i}$. The cost in this case is $i \times 1 + (\nu - i)D$, so the random cost is the sum of two binomial random variables. The average equals $E = \nu(1 - p + Dp)$. Violated deadline means penalty F so we add the expected penalty: $E = \nu(1 - p + Dp) + p^\nu F$.

To minimize it consider the derivative: $E' = (1 - p + Dp) + p^\nu \ln pF$, so ν^* satisfies

$$p^\nu = \frac{1 - p + Dp}{-\ln pF}.$$

A necessary condition for $\nu > 1$ is $-\ln pF > 1 + (D - 1)p$. Note that again critical penalties are reciprocal of the risk and that the relation between F and p is also logarithmic.

Now assume that the response time is distributed with some CDF f and also that a node does not respond at all with probability $\bar{p}$. Then probability p of missing the deadline D is $p = \bar{p} + (1 - f(D))(1 - \bar{p})$. This allows to choose the optimal deadline together with the optimal replication provided that the distribution function f is known.

Let us consider the case $\bar{p} = 0$. If $(1 - f(D))D \to 0$ as $D \to \infty$ (the case for most popular distributions) then the optimal deadline is infinite; this means that the deadline is useless, it is reasonable to just wait for an answer paying, in average, the mean unit cost. However, for the Cauchy distribution $(1 - f)D \to 1$ so that $E \to 2$; in the exotic case of a CDF $f(D)$ such that $D(1 - f(D)) \to \infty$ some finite deadline is still necessary. However, the usual case for a Desktop Grid is $\bar{p} > 0$: nodes may leave the project, temporarily of permanently.

Now assume that we have M increasing deadlines D_j, $j = 1, \ldots, M$, and probability to violate the deadline j is p_j. Add artificial deadlines $D_0 = 0$ and $D_{M+1} = D_M$. In fact, the last deadline is infinite, but if the D_M is violated, the task is cancelled. The sequence p_j decrease with respect to j, $p_0 = 1$ (no task can be returned instantly), $p_{M+1} = 0$ (infinite deadline can not be violated). Violation of the deadline D_j costs the penalty F_j; penalties increase with respect to j and let $F_1 = 0$ (no penalty for quick work).

Then the average cost is

$$E = \nu \sum_{j=1}^{M+1} p_{j-1}(1 - p_j)D_j + \sum_{i=1}^{M+1} (1 - p_i^\nu)p_{i-1}^\nu F_i$$

This function has a minimum for a $\nu \geq 1$ because for large ν it grows almost linearly: the second term tends to zero quite quickly as $\nu \to \infty$.

2.3 The reliable computer and penalty/cost estimations

Let us estimate the penalty using a reliable computer with $p = 0$ but high cost C of usage. The set of answers is binary (yes/no) with no information on their probabilities. It is clear that if this computer is better (from the point of view of the mean total cost) than the Desktop Grid, the latter is just useless.

So assume that $p = 0.5$ which means that tossing a coin is as good as solving tasks. Then the expected penalty is $0.5F$ which must exceed C: otherwise tossing a coin is better than solving all tasks on reliable nodes. So $F > 2C$ is a necessary condition for choosing the penalty value.

If there are S possible answers, the estimation becomes $(S-1)F > SC$. In case of known probabilities α_i of answers tossing a coin is replaced by guessing the most likely answer (with probability Q) yielding the estimate $(1-Q)F > C$. Indeed, choosing a distribution x_i to guess an answer in order to maximize the probability to guess right we have the linear optimization problem

$$\sum \alpha_i x_i \to \max, \quad \sum x_i = 1, \quad x_i \geq 0.$$

Assume that α_i decrease and exclude the x_1 to get the linear problem

$$1 + \sum (\alpha_i - \alpha_1) x_i \to \max, \quad \sum x_i \leq 1, \quad x_i \geq 0.$$

With negative constant gradient, the function increases up to the boundary so that one of the variables vanishes; continue to eliminate all variables except $x_1 = 1$.

Note than the uniform distribution is the entropy-maximizing solution for distribution on the finite set of points with no restrictions. In case of countable set such a solution does not exist; however, fixing the expectation $W > 0$ of the distribution provides the solution which is the geometrical distribution $P(1-P)^i$ on $i = 0, 1, \ldots$, where

$$P = \frac{1}{W+1}.$$

The discussed estimation is $(1-P)F > C$.

Now let us obtain the other estimation. In case of relatively reliable computing nodes the optimal cost must be less than C: otherwise there is no need to use the Desktop grid. If p is small so that p^2 is negligible with respect to p, then the wrong result is accepted with probability

$$p^\nu \sum_{i=0}^{\nu-1} \binom{\nu+i-1}{\nu-1} = p^\nu \binom{2\nu-1}{\nu-1} = \frac{p^\nu}{2} \binom{2\nu}{\nu} \sim \frac{4^\nu}{\sqrt{\pi\nu}}.$$

The average cost without the penalty and discarding terms with p is ν: most times we get ν correct answers in a row. So the reliable cost must be high enough to make using the Desktop Grid reasonable:

$$\nu + \frac{(4p)^\nu}{\sqrt{\pi\nu}} F < C$$

for the optimal ν (equivalently, for at least some ν). As p is small, large ν mean very quickly decreasing second term. So if the penalty F is so high that the inequality does not hold for any $\nu < C$, the Desktop Grid is useless.

These two estimates allows to choose the penalty value (if it is unknown) to solve the optimal replication problem.

Another approach is able to estimate the cost C. Assume for the sake of generality that there are many groups of nodes with their own costs C_i and risks p_i; also assume that tasks are replicated within each group. Then each group possesses its own average total cost $E_i = E(C_i, \nu_i, p_i, F) > 0$ provided by its own quorum ν_i. The penalty for accepting a wrong answer F is the same for all groups. If we try to determine relative amounts ψ_i of tasks sent to each group in order to minimize the cost, we get a simple linear optimization problem

$$T = \sum_i \psi_i E_i \to \min, \quad \sum_i \psi_i = 1, \quad \psi_i \geq 0.$$

Without loss of generality assume that E_i increase with respect to i. This problem has a solution $\psi_1 = 1$, other $\psi_i = 0$, so all tasks are computed on the cheapest (with replication taken into account) node. Indeed, express ψ_1 from the constraint and substitute to expression for T:

$$T = E_1 + \sum_i \psi_i (E_i - E_1) \to \min, \quad \sum_{i>1} \psi_i \le 1, \quad \psi_i \ge 0.$$

The gradient of T is positive, so the descent eliminates all ψ_i, $i > 1$, making $\psi_1 = 1$. Note that the similar argument remains true in a rather general nonlinear case.

However, high load of a node would generally increase the cost per a task just because more tasks need to be solved. Then the optimal ψ_i are such that all costs with $\psi_i > 0$ are equal and costs with $\psi = 0$ are higher. This means that hopeless nodes are excluded while nodes with acceptable cost/reliability are loaded in such a way that the average total cost is the same. This allows to estimate the basic cost of computing for a task. If nodes are choosing their ψ_i and are paid the same T, this is the congestion game; such games were applied to scheduling in [NIT17]. Game approach allows to estimate the cost in a decentralized way.

3 Groups of tasks

Formation of task parcels containing many tasks and considered as a single task can be useful for Desktop Grid computing. We faced the problem of loss of performance due to much additional work when the Desktop Grid solved many vary quick tasks, so that the server lost much time to processing reports and task retrieval requests. Grouping tasks into parcel with 1000 simple tasks in each improved the performance significantly.

3.1 Reducing additional costs

Let us estimate the reasonable batch size depending on reliability of computing nodes and additional costs. Denote the parcel size by n tasks including N tasks from other parcels. Cost for evaluating a task is unit: it includes the cost (or time) of uploading a task and all necessary information. Let c be the cost of additional work for the whole parcel. Assume that probability of solving a task is q (so $p = 1 - q$ is for the probability of a fault) and that tasks are solved independently. Then probability of processing the whole parcel with no errors is q^n. In case of a fault the whole parcel needs to be solved again. Solving a parcel in k attempts costs $(n + c)k$ units and has the probability $(1 - q^n)^{k-1} q^n$. The expected cost of solving a parcel is

$$E_n = \frac{n + c}{q^n}.$$

Solving all tasks costs approximately N/n times more which gives us the total average cost per a task:

$$E = \frac{N}{n} \frac{E_n}{N} = \frac{n + c}{nq^n}. \tag{1}$$

Find the minimal value of this cost:

$$E' = \frac{nq^n - (n + c)(q^n + nq^n \ln q)}{n^2 q^{2n}} = \frac{n - (n + c)(1 + n \ln q)}{n^2 q^n} = 0.$$

This equation is reduced to

$$An^2 + Acn - c = 0$$

where $A = -\ln q$. This square equation has the single positive root

$$n^* = \frac{\sqrt{A^2 c^2 + 4Ac} - Ac}{2A}.$$

For example, for $c = 1$, $p = 10^{-2}$ we have $n^* = 9$.

Note that n^* does not depend on the number N of tasks added from other parcels. Consider inequality $n^* > 1$ to see when using parcels is reasonable. It is only if $q > \mathrm{e}^{-1}$ and

$$c > \frac{A}{1 - A}, \quad A = \ln \frac{1}{q}.$$

In other words, grouping can help if nodes are reliable enough and additional costs are high enough.

For reliability reported at the BOINC web site ($q \approx 0.95$) grouping seems reasonable for similar additional cost with respect to the average cost of a single task: $c > 5.4\%$.

3.2 Embedding test tasks

Let us assume that tasks are sent to computing nodes in parcels of N tasks each; in order to reduce the risk of accepting a wrong answer, k answers of each parcel are replicated by being included into other parcels. If at least one replicated task produce different answers, all parcel is solved again. As the first approximation, we assume that a parcel solved again is solved correctly. Such embedding test tasks into parcels can be called "fractional replication": a part of each parcel is replicated.

A wrong answer can be obtained with some probability p, probability of the right answer is $q = 1 - p$.

If a wrong tasks is believed, some penalty F is added to the overall computation cost. The cost function is additional time spent for checks and penalties.

Then there is no alarm if both replicas of replicated tasks produce the same answer, either right or wrong; in other words, probability of the "no alarm" case is

$$\sum_{i=0}^{k} \binom{k}{i} (p^i q^{k-i})^2.$$

All terms except the $i = 0$ contain errors and, therefore, penalties. The following cases are possible:

1. No errors; additional cost is k, probability is q^{N+k}.
2. An error, all control tasks solved correctly; cost is $k + F$, probability is $q^{2k}(1 - q^{N-k}) = q^{2k} - q^{N+k}$.
3. An error, no alarm; cost is $k + F$, probability is $\sum_{i=1}^{k} \binom{k}{i} (p^i q^{k-i})^2$.
4. An error detected; cost is $k + N$, probability is $1 - \sum_{i=0}^{k} \binom{k}{i} (p^i q^{k-i})^2$.

So the average cost is

$$E_N^k = k + F\left(q^{2k} - q^{N+k} + \sum_{i=1}^{k} \binom{k}{i} p^{2i} q^{2(k-i)}\right) + N\left(1 - \sum_{i=0}^{k} \binom{k}{i} p^{2i} q^{2(k-i)}\right).$$

This quantity increases with respect to N for a fixed k; as $k \leq N$, then the optimal N equals k. This means that no grouping is necessary for optimizing the cost; however, the parcel size N can be fixed due to other reasons, as we have noted above. Then the solution to the optimization problem is $N = k$, which means the complete replication (duplication) of a whole parcel.

These preliminary estimations seem promising for the improved task-parcel management to be developed.

Conclusion

The mean-cost approach has been shown here to be effective not only for improving performance by replication, but also to increase sabotage-tolerance, decrease deadline-violation losses, and effectively use nodes of various reliability/cost ratio. Penalties and costs can be a priori estimated if not known precisely. Task grouping (formation of task parcels) can serve efficiency due to less additional costs on client-server interaction and lower server load if computing nodes are reliable while the losses are relatively high. Estimations for these quantities are presented. Adding test tasks to parcels, though possibly useful, can hardly be used as a "partial replication" method if the only optimization criteria is the mean cost.

Acknowledgements

This work is supported by the Russian Foundation for Basic Research (grant number 16-07-00622).

References

[Che16] I. Chernov. Theoretical study of replication in desktop grid computing: Minimizing the mean cost. In *Proceedings of the 2nd Applications in Information Technology (ICAIT-2016), International Conference on*, pages 125–129, Aizu-Wakamatsu, Japan, 2016.

[CKBH06] S.J. Choi, H.S. Kim, E.J. Byun, and C.S. Hwan. A taxonomy of desktop grid systems focusing on scheduling. Technical report KU-CSE-2006-1120-02, Department of Computer Science and Engeering, Korea University, 2006.

[CN15] I.A. Chernov and N.N. Nikitina. Virtual screening in a desktop grid: Replication and the optimal quorum. In V. Malyshkin, editor, *Parallel Computing Technologies, International Conference on*, volume 9251, pages 258–267. Springer, 2015.

[DS14] N.M. Durrani and J.A. Shamsi. Volunteer computing: requirements, challenges, and solutions. *Journal of Network and Computer Applications*, 39:369–380, mar 2014.

[ET12] T. Estrada and M. Taufer. Challenges in designing scheduling policies in volunteer computing. In C. Cérin and G. Fedak, editors, *Desktop Grid Computing*, pages 167–190. CRC Press, 2012.

[KMH17] M.Kh. Khan, T. Mahmood, and S.I. Hyder. Scheduling in desktop grid systems: Theoretical evaluation of policies and frameworks. *International Journal of Advanced Computer Science and Applications*, 8(1):119–127, 2017.

[NIT17] N. Nikitina, E. Ivashko, and A. Tchernykh. Congestion game scheduling for virtual drug screening optimization. Submitted to The Journal of Chemical Information and Modeling, 2017.

[XA10] F. Xhafa and A. Abraham. Computational models and heuristic methods for grid scheduling problems. *Future Generation Computer Systems*, 26(4):608–621, apr 2010.