Balancing distributed analytics' energy consumption using physics-inspired models

Brent Kraczek, Theodoros Salonidis, Prithwish Basu, Sayed Saghaian, Ali Sydney, et al.

Brent Kraczek, Theodoros Salonidis, Prithwish Basu, Sayed Saghaian, Ali Sydney, Bongjun Ko, Tom LaPorta, Kevin Chan, James Lambert, "Balancing distributed analytics' energy consumption using physics-inspired models," Proc. SPIE 10652, Disruptive Technologies in Information Sciences, 1065206 (9 May 2018); doi: 10.1117/12.2304485

SPIE.

Event: SPIE Defense + Security, 2018, Orlando, Florida, United States

Balancing distributed analytics' energy consumption using physics-inspired models

Brent Kraczek[a], Theodoros Salonidis[b], Prithwish Basu[c], Sayed Saghaian[d], Ali Sydney[b], Bongjun Ko[c], Tom LaPorta[d], Kevin Chan[e], and James Lambert[f]

[a]US Army Research Laboratory, 328 Hopkins Rd., RDRL CIH-C, Aberdeen Proving Ground, MD 21005, USA
[b]Raytheon BBN Technologies, 10 Moulton Street, Cambridge, MA 02138, USA
[c]IBM T.J. Watson Research Center, 1101 Kitchawan Rd, Yorktown Heights, NY 10598, USA
[d]School of Electrical Engineering and Computer Science, The Pennsylvania State University, 207 Electrical Engineering West, University Park, PA 16802, USA
[e]US Army Research Laboratory, 2800 Powder Mill Rd, RDRL CIN-T, Adelphi, MD 20783, USA
[f]Defence Science and Technology Laboratory, Porton Down, Salisbury SP4 0JQ, UK

ABSTRACT

With the rise of small, networked sensors, the volume of data generated increasingly require curation by AI to analyze which events are of sufficient importance to report to human operators. We consider the ultimate limit of edge computing, when it is impractical to employ external resources for the curation, but individual devices have insufficient computing resources to perform the analytics themselves. In a previous paper we introduced a decenralized method that distributes the analytics over the network of devices, employing simulated annealing, based on physics-inspired Metropolis Monte Carlo. If the present paper we discuss the capability of this method to balance the energy consumption of the placement on a network of heterogeneous resources. We introduce the *balanced utilization index* (BUI), an adaptation of Jain's Fairness Index, to measure this balance.

Keywords: Distributed analytics; simulated annealing; load balancing

1. INTRODUCTION

The current proliferation of small, cheap, network-connected devices, ranging from cell phones to internet of things (IoT) devices, means that the number of cameras, sensors and small processors is increasing exponentially. For example, one investiment firm estimates that by the year 2022 there will be 45 billion cameras worldwide and that the vast majority of the pictures taken by these cameras will only ever be "viewed" by AI.[1] With such a flood of images, the prevalent practice of sending the data to a centralized data center for storage and processing is increasingly impractical, especially given that most data generated will be "uninteresting" and have limited value. This increase in data generated has led to the rise of *edge computing*, in which small "microservers" are used to process data locally, determining what data to send to large data centers. While this is ideal in controlled environments, the reliance on a single point of failure is not ideal in hazardous or exposed environments, such as in military and mining applications. We consider a different approach–the distribution of the analytics across the deployed devices themselves. In order to keep costs as low as possible, an individual device may not have sufficient resources to perform the analytics locally. Instead, we consider the case that the distribution of such analytics across the processing capabilities of the networked devices is enabled by the ubiquity of high-speed, low-power wireless communications.

As a use-case scenario, we consider an array of cameras and sensors used in a security or surveilance application. The devices would built on a cheap, cell-phone-like platform, including WiFi, a processor, small amounts of

Further author information: (Send correspondence to B.K.)
B.K.: E-mail: brent.e.kraczek.civ@mail.mil

Disruptive Technologies in Information Sciences, edited by Misty Blowers, Russell D. Hall, Venkateswara R. Dasari
Proc. of SPIE Vol. 10652, 1065206 · © 2018 SPIE · CCC code: 0277-786X/18/$18 · doi: 10.1117/12.2304485

RAM and persistent storage, a battery and a solar cell. The devices could consist of different sensor types and/or different hardware generations, making the computing system heterogeneous. In addition, for any given event of interest, it is likely that the system load will be non-uniform, with one to a few sensors being primarily engaged. The robustness of such a system will depend on the distributed algorithms that coordinate the analytics, but ensure there is no single point of failure: any given node of the system might fail, run out of energy, or be subject to a physical or malicious attackers, but the system can still perform the needed analytics.

In this paper we address the problem of placement of analytics graphs in a dsitributed wireless network. We view this problem as a graph embedding problem, which is typically intractable.[2,3] Our approach uses a physics-inspired formulation which is also amenable to a distributed solution. This problem is addressed using two key components. 1. Interactions between resource types are inspired by physics-like interactions, such as spring forces for communications and Coulomb-like forces to balance the distribution of code and data "particles". 2. An objective function is defined and optimized using simulated anneaing (SA),[4] based on the Metropolis Monte Carlo (MMC) method commonly used for atomic-scale simulation,[5,6] denoted below as MMCSA. Consistent with the scenario described above, we call the analytics graph the *logical graph*, embedded within the *physical graph*, set by the network of sensor devices.

The use of physics models and physics-inspired models for network optimization has been addressed by a number of groups.[7] For example, Adamic *et al.*[8] have demonstrated that power-law graphs can enhance search algorithms for networks that include a few nodes with a high degree of connectivity. More recently, Yeung *et al.* have solved the network mapping problem for sparse graphs using statistical physics approaches.[9] However, none of these approaches use such physics-inspired formulations for distributed analytics placement. In our recent work,[10] we considered a simplified version of the present problem and evaluated the efficacy of three types of physical forces to model the movement of agile code and data objects, namely gravity, elastic and a Coulomb-like interaction. While each has its pros and cons in modeling our problem, in our previous work we only applied them individually and not together. In this paper, we apply both the elastic and the Coulomb-like force models simultaneously, as done in molecular modeling, to model the various types of interactions, as explained in section 3.

This paper provides a more realistic model, combining different types of forces, presents the MMCSA algorithm for solving it, and focuses on the ability of this algorithm to balance compute energy consumption. This load balancing is obtained through the use of Coulomb-like interactions that simultaenously act as penalty functions to ensure the analytics placement does not exceed (violate) the processing energy capacity of any given physical node. While the standard fairness measure, Jain's Fairness Index (JFI),[11] is appropriate to measure this balancing in the case of homogeneous resources, it does not address the case of heterogeneous resources. The Coulomb-like interactions between logical graph particles naturally balance the energy utilization, where the capacities after computation is uniform. To measure this, we introduce an adaptation of JFI, which we call the *balanced utilization index* (BUI). Finally, while the implementation used to generate the results in[10] and the current paper is not distributed, it is constructed such that it can be employed in a decentralized or distributed manner. This is possible because the objective function in the SA optimization is additive, such that only the change in the change in objective function is needed to determine whether to accept a trial move. This renders the algorithm consistent with atomistic algorithms that are regularly used in distributed computations,[12] provided that all interactions between code and data nodes of the logical and physical graphs are small in number (as in communications) or finite-ranged (as in the Coulomb-like interactions).

The balance of the paper is structured as follows. In section 2 we discuss how the MMCSA method given here balances resource utilization, and define the BUI as a measure of this utilization. In the third section we give results from experiments of the method, which demonstrate its performance and the tuning the objective function causes a transition between a communications-optimized (low communications energy, low BUI) and a utilization-optimized (high communications, high BUI) result. We conclude with a discussion of future directions for this research.

2. MMCSA METHOD FOR ANALYTICS PLACEMENT

We define the logical (analtyics) graph $\mathcal{G} = (V_A, E_A, q, r, \omega_A^E)$, composed of nodes/vertices V_A and edges E_A, with weights $q : V_A \to \mathbb{R}^+$, $r : V_A \to \mathbb{R}^+$, $\omega_A^E : V_A \times V_A \to \mathbb{R}_{\geq 0}$, set respectively as the constant processing energy cost of each analtyics stage, RAM requirement of each analtyics stage and the directed data transmission (in megabytes) between each pair analytics stages (the majority of which will be zero). Similary, we define the physical (resource) graph $\mathcal{H} = (V_P, E_P, Q, R, \omega_P^E)$, composed of composed of nodes/vertices V_P and edges E_P, with weights $R : V_P \to \mathbb{R}^-$, $Q : V_P \to \mathbb{R}^-$, $\omega_P^E : V_P \times V_P \to \mathbb{R}^+$, set respectively as the processing energy capacities and RAM of each physical node and the data transmission cost (in Joules per megabyte) between physical nodes. The placements of individual logical nodes on the physical graph are given by $\pi : V_A \to V_P$, and are updated throughout the calculation.

2.1 Objective function

We employ an objective function consisting of three primary terms,

$$\Phi = \Phi_{\text{comms}} + c_{\text{RAM}}\Phi_{\text{RAM}} + c_{\text{proc}}\Phi_{\text{proc}} \ , \tag{1}$$

each of a different type.

1. For energy costs due to communications, which we desire to minimize:

$$\Phi_{\text{comms}} = E^{\text{comms}} \ , \tag{2a}$$

$$E^{\text{comms}} = \sum_{(u,v)\in E_A} \omega_A^E\,(u,v)\,\omega_P^E\,(\pi(u), \pi(v)) \ . \tag{2b}$$

2. To ensure that the RAM requirements of the analytics task(s) placed on a physical node do not exceed the available resources, Φ_{RAM}acts as a penalty term,

$$\Phi_{\text{RAM}} = \sum_{z\in V_P} \mathbb{1}\left\{R_z + \sum_{u\in V_A} r_u \mathbb{1}\,\{\pi(u) = z\} > 0\right\} \ , \tag{3}$$

which employs the indicator function $\mathbb{1}$.

3. For the energy requirements due to processing, we could employ an indicator function like that in Φ_{RAM}. However, as discussed below, it will often be desirable to distribute the processes so that the energy usage is balanced among the physical nodes. For this reason, the contribution Φ_{proc}employs a Coulomb-like interaction that both penalizes overcapacity and balances the resource placement:

$$\Phi_{\text{proc}} = \sum_{z\in V_P}\sum_{u\in V_A} Q_z q_u \ f\,(z, \pi(u)) + \sum_{(u,v)\in V_A\times V_A} q_u q_v \ f\,(\pi(u), \pi(v)) \ , \tag{4}$$

where $f : V_P \times V_P \to \mathbb{R}_{\geq 0}$ is a non-singular function of the distance between physical nodes, replacing the $1/r$ potential familiar to the Coulombic interaction. In the current implementation, we employ a polynomial form for $f(u, v)$ that is a finite ranged, non-increasing, non-negative function of the same multi-hop distance employed by Φ_{comms}. The form of (4) has the natural effect of causing repulsion between logical compute nodes, while attracting the largest (positive) logical charges to the minimum (largest-magnitude negative) physical charges. Given the nature of the problem, physical node placement is fixed prior to calculations of Coulomb-like interaction.

It is important to note that (1) is non-convex in general, and that the individual terms represent three different objectives. The constants c_{RAM} and c_{proc} are chosen heuristically to balance these objectives, and relate to the energy scale in Φ_{comms} and the temperature schedule τ employed by the MMCSA algorithm. Provided that the RAM load of a logical graph is sufficiently smaller than total RAM capacity of the physical graph, and that the RAM requirements of individual nodes of the logical graph are smaller than the individual physical node capacities, the parameter c_{RAM} can generally be determined and remain fixed for a specific set of calculations. In addition to ensuring that the processing energy capacities are not exceeded, c_{proc} also can be varied such that the optimization favors minimizing communications energy cost or favors maximizing the BUI, a possibility we explore in the numerical experiments below.

2.2 MMCSA Algorithm

We employ MMCSA to obtain the set of placements on the nodes of the physical graph that minimize the objective function Φ. Following standard MMCSA procedure (algorithm 1), we employ a temperature τ, which follows a cooling schedule. Because the system is not physical motivated, τ and the cooling schedule are selected based on the scenario under consideration. In the current implementation we employ an exponential function for τ that decays from τ_{init} at $s = 0$ (initialization) to τ_{min} at the MC step $s = s_{\text{max}}$. At each MC step s, a logical node is chosen at random and subjected to a trial move on the physical graph. With each trial move, if the change $\Delta\Phi$ is negative, the trial is accepted automatically. If $\Delta\Phi$ is positive it is accepted if $\exp(-\Delta\Phi/\tau) > \text{rand}(0, 1)$.

Algorithm 1 MMC method for optimizing "logical" graph placement on a "physical" graph. The algorithm is MMC for a fixed τ and MMCSA for τ that varies with s.

```
Set physical graph within a 2D Euclidean space.
Embed logical graph(s) in physical graph, with intermediate nodes distributed evenly between endpoints.
Calculate $\Phi$ for initial configuration.
for $s = 1 : s_{\max}$ do
    Randomly choose loical graph node $u$.
    Displace node $u$ from physical graph node $z$ to trial node $z'$.
    Compute $\Delta\Phi \leftarrow \Phi(\pi(u) = z') - \Phi(\pi(u) = z)$.
    if $\Delta\Phi < 0$ then
        Set $a \leftarrow$ True.
    else if $e^{-\Delta\Phi/\tau(s)} > \text{rand}(0, 1)$ then
        Set $a \leftarrow$ True
    else
        Set $a \leftarrow$ False
    end if
    if $a$ then
        $\Phi \leftarrow \Phi + \Delta\Phi$
        $\pi(u) \leftarrow z'$
    end if
    Record data
end for
     return Placement of logical graph nodes $\{\pi u\}$.
```

The evolution of the system during the MMCSA method is the search for the optimal solution and not the actual evolution of the system, so it is only important that the soft constraints, handled by the penalty terms Φ_{proc} and Φ_{RAM}, are not violated in the final solution. It is often beneficial during the MMCSA calculation that both the RAM and processing energy capacities are exceeded early in the calcuation, when values of τ are high, to discourage trapping of the solution in a less-optimal, local minimum. Naturally, the choice of cooling schedule $\tau(s)$ determines the quality of the final configurations obtained by the MMCSA calculations, while sensible initial placements of the logical graph nodes on the physical graph can decrease the number of iterations required.

2.3 Capacities and the balanced utilization index (BUI)

The choice of processing capacities, $\{Q_z\}$ should be motivated by the specific problem being solved, but can be modified to bias the placement toward additional objectives. For devices relying on battery power, attention must be paid to ensuring that devices do not run out of energy entirely, and the $\{Q_z\}$ can be chosen based solely on this objective. In the security camera example given above, a robust system will seek to keep all cameras operational for as long as possible. In many cases, the operator will not know *a prior* where the largest energy costs will occur on the physical graph throughout a series of computations. In this case the safest placement scheme is to optimize for uniform final capacity on the graph after each calculation. Balanced utilization in this case would assign values of set of physical resources $\{Q_z\}$ based on available power at the beginning of the calculation. Alternatively, the network operator may choose to bias calculations toward certain resources, motivated by the relative reliabilities of resources, the need to keep some resources in reserve or some other criterion. In this case certain values in $\{Q_z\}$ would be modified from the initial capacities, with values at specific physical nodes increased (decreased in magnitude) to bias the utilization away from them.

In order to evaluate how well this algorithm attains balance across heterogeneous resources, we have adapted JFI to reflect how well this method balances the final available processing energy, which at a single physical node is given by

$$\bar{Q}_z = Q_z + \sum_{u \in V_A} q_u \mathbb{1}\{\pi(u) = z\} \ . \tag{5}$$

We begin with the recognition that the JFI assumes that all contributions are non-negative, ensuring that the measure monotonically increases as the "fairness" increases. We can guarantee that all contributions are non-negative by subtracting the minimum (largest magnitude) physical charge, $Q_{\min} = \min(\{Q_z\})$, from each contribution,

$$Q'_z = \bar{Q}_z - Q_{\min} \ . \tag{6}$$

Employing Q'_z in the JFI, we get the BUI,

$$I_{BU} = \frac{\left(\sum_{z=1}^{n} Q'_z\right)^2}{n \cdot \sum_{z=1}^{n} \left(Q'_z\right)^2} \ . \tag{7}$$

The possible limiting values of the BUI are the same as the limits if JFI, from $1/n$ to 1, though, as with the JFI, the actual limits are dependent on physical and logical graph. When the initial capacities are uniform, $\{Q_z\} - Q_{\min} = 0$ and the BUI reduces to JFI in which only the distribution of positive charges is considered.

3. EXPERIMENTS

In this section we present the results of a series of numerical experiments to demonstrate the capabilities of the MMCSA method in balancing the analytics placement on a network of similar devices. For the logical graph, we simulate the distributed use of one or more instances of the AlexNet image processing algorithm,[13] with the 10 layers of the algorithm distributed on separate nodes of the logical graph. The initial and final points of the logical graph are fixed on the physical graph, representing an observing sensor and a designated node for transmission from the network to an outside observer; the intermediate nodes of the logical graph are moved by the MMCSA method. The energy costs are based on 100 evaluations of AlexNet on a Samsung Galaxy S5 smartphone, though with the RAM limited to 256MB per node. For communications, we define the Euclidean positions of the physical nodes and determine $\omega_P^E : V_P$ based on the Euclidean distances between the nodes with the energy costs derived from the minimum multi-hop energy costs under the 802.11ac 20MHz DS 2x2 MIMO wireless spec, as measured by Saha *et al.*[14] Numerical experiments demonstrate that with this configuration, the parameters employed within the MMCSA method can be readily determined and are robust across different physical configurations. In order to demonstrate the efficacy of the MMCSA method in balancing resource utilization, the processing energy capacities of physical nodes were chosen to be non-uniform and represent a system with enough resources to process roughly two sets of images.

3.1 Experiment 1: Evolution of simple analytics placement

For the physical graph, we begin with a geometric random graph with 16 nodes, distributed in a square kilometer. (In our numerical testing, 16 nodes is about the minimum number of physical nodes required to guarantee that a random distribution of nodes is fully-connected on a square kilometer at the maximum range for the internodal communications assumed in ω_P^E.) In this experiment the logical graph consists of a single instance of the AlexNet algorithm. In figure 1 we show schematics of the processing energy and RAM for the initial ($s = 0$) and final ($s = 5000$) configurations, as well as one intermediate configuration ($s = 2000$). The initial condition was chosen to lie on a short path between the initial and final positions, with the analytics positioned along this path. The areas of blue left-semicircles are proportional to the available capacity for RAM or processing energy for the physical node located at the semicircle's origin. The areas of the opposing, right-semicircles represent the size-proportional utilization. These are colored red at nodes that violate the capacities and green at those that are within capacity. The gray lines connecting the physical nodes show the available WiFi connections, based on the distances employed in ω_P^E. The heavy colored lines show which connections are used for communications by the distributed analytics calculation, with different colors for distinct analytics calculations. A circle represents the point where the sensor makes an observation, and thus starts a calculation, represented by the heavy lines of the same color; the black triangle is the point where the final analysis is delivered. The initial placement of logical nodes distributes the nodes as uniformly as possible along the lowest-cost communications pathway between the observation point and the final point. As can be seen in the schematics in figure 1, the initial placements contain violations of both the processing energy and RAM capacities. After running the MMCSA algorithm the placements satisfy the capacities on the system. It is important to note that, for a value of $c_{\text{RAM}} = 1000$ we have observed no violations of RAM capacities for any of the calculations represented in this paper, so we omit any representation of it in the subsequent results. In figure 2 we show the evolution of E^{comms} and the BUI of the processing energy as a function of the MMCSA iteration number. Thus both the communications energy and the BUI have relatively small initial values. The cooling schedule is a simple, decaying exponential. Thus, the both quantities quickly increase and fluctuate wildly in the initial "high-temperature" portion of the annealing. As the temperature decreases, the primary contributions to the objective function become fixed, followed by the lesser contributions.

3.2 Experiment 2a: Impact of Network Size

Next, we demonstrate the efficacy of the algorithm for a physical graph with 80 nodes on four square kilometers, with five separate AlexNet calculations in the logical graph. The different AlexNet calculations have different initial points (colored circles) and a single final point (the black triangle), with the associated communications of the same color as the initial points. The initial points are chosen such that we expect congestion, including one initial point that is very near to the final point, and a region of the graph in which there are no initial points, so that we might reasonably expect that it will be less-utilized. In figure 3 we show the final placement for a single configuration of this placement, as well as the the evolution of E^{comms}) and the BUI as a function of MC step s. Similar to the results shown in figures 1 and 2, the MMCSA method finds a placement of analytics stages that balances the energy usage while also employing relatively efficient communications pathways.

3.3 Experiment 2b: Impact of balancing parameter

In order to understand the role of the parameter c_{proc} in balancing the communications and processing energies, we employ this 80-physical-node graph plus logical graph consisting of 5 AlexNet computations and associated initial conditions, but now use 51 values for c_{proc}, ranging from c_{proc}=20 to c_{proc}=32000, and spaced roughly logarithmically. For each value of c_{proc} we ran 100 MMCSA calculations with different random seeds. In figure 4 we plot the average values for E^{comms}, the BUI and the number of violations of the processing capacity (i.e. when $\bar{Q}_z > 0$) as a function of c_{proc}. As seen in this figure, the values of both E^{comms} and the BUI increase monotonically, to within statistical noise, with c_{proc}. Similarly, we find that the number of processing capacity violations decreases with c_{proc}, until for $c_{\text{proc}} > 6000$ the probability of a single capacity violation is approaching zero, such that it would be reasonable to use this parameter. In this case the values c_{proc} could be chosen above 6000 to optimize for higher BUI or lower E^{comms}. However, given that around this value for c_{proc} E^{comms} begins to increase more rapidly, while BUI's increases begin to slow, it may not be desirable to use values much above c_{proc}=10000.

3.4 Experiment 3: Impact of processing capacity

In the final experiment, we investigate the effects of processing capacity by moderately increasing the available processing capacities from the configuration employed in experiments 2a and 2b. Qualitatively, the physical graph configuration employed in experiments 2a and 2b employ 40 "large-capacity", 20 "mid-capacity", and 20 "small-capacity" physical nodes. Experiment 3 is obtained from experiment 2b by changing the 20 mid-capacity nodes to large-capacity nodes. As shown in figure 5, the trends of the average values for E^{comms}, the BUI and the number of violations of the processing capacity are consistent with the previous figure. However, the average number of processing violations drops to zero at a lower value. Thus, in this less-constrained case, there is more flexibility in using c_{proc} to balance optimization favoring E^{comms} or the BUI.

4. CONCLUSION AND FUTURE WORK

In this paper we have demonstrated a physics-inspired MMCSA algorithm for distributed analytics placement on a random graph and evaluated its efficacy in the optimizing this placement. This is done by formulating the problem as a graph embedding problem and designing the objective function to employ penalty functions to enforce system capacities and balance system utlization. We have also introduced the balanced utilization index (BUI), an extension of Jain's Fairness Index, that provides a metric of how well different placements balance the remaining energy for processing, after the analytics calculations are completed.

In the case of the RAM capacities of the system, the penalty term Φ_{RAM}, based on indicator functions, is sufficiently effective that we observed no violations of this capacity in any of the calculations represented in this paper. Φ_{proc}, the Coulomb-like term in the objective function plays a dual role. As with the Φ_{RAM}, it penalizes over-capacity placements of processing in the analytics stages. It also has the effect of balancing the placement of the analytics stages in two ways. It balances the utilization of the processing energy capacity, as measured by the BUI. It also serves as a tuning parameter, allowing shifting the emphasis of the optimization between communications energy and balance of remaining processing energy. This becomes more effective as the total capacity of the graph increases.

In the future, we anticipate that the performance of the method, both in terms of decreasing the number of iterations required and the optimality of solutions, could be improved by choosing more complicated cooling schedules. We speculate this could be achieved by plateauing τ at targeted temperatures to better relax the placment of specific portions of the logical graph. A second consideration will be to simulate the effect of adding logical graphs sequentially, simulating a different use-case. Finally, we anticipate that with a modified objective function, this method could be applied to the optimization other network topologies, including those of heterogeneous computing platforms, such as those composed of many CPUs and GPUs.

ACKNOWLEDGMENTS

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the ofcial policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

REFERENCES

[1] Nisselson, E., Hunter-Syed, A., and Shah, S., "45 billion cameras by 2022 fuel business opportunities," tech. rep., LDV Capital, New York, NY (2017).

[2] Bokhari, S. H., "On the mapping problem," *IEEE Trans. Comput.* **30**, 207–214 (Mar. 1981).

[3] Monien, B. and Sudborough, H., [*Embedding one Interconnection Network in Another*], 257–282, Springer Vienna, Vienna (1990).

[4] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by simulated annealing," *Science* **220**(4598), 671–680 (1983).

[5] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E., "Equations of state calculations by fast computing machine," *J. Chem. Phys.* **21**, 1087–1091 (1953).

[6] Frenkel, D. and Smit, B., [*Understanding Molecular Simulation*], Academic Press, Inc., Orlando, FL, 2nd ed. (2002).

[7] Saad, D., Yeung, C. H., Rodolakis, G., Syrivelis, D., Koutsopoulos, I., Tassiulas, L., Urbanke, R., Giaccone, P., and Leonardi, E., "Physics-inspired methods for networking and communications," *IEEE Communications Magazine* , 144–151 (November 2014).

[8] Adamic, L. A., Lukose, R. M., Puniyani, A. R., and Huberman, B. A., "Search in power-law networks," *Phys. Rev. E* **64**, 046135 (2001).

[9] Yeung, C. H. and Saad, D., "Competition for shortest paths on sparse graphs," *Phys. Rev. Lett.* **108**, 208701 (2012).

[10] Ko, B., Kraczek, B., Salonidis, T., Basu, P., Chan, K., Porta, T. L., Martens, A., and Lambert, J., "Proc. 1st workshop on distributed analytics infrastructure and algorithms for multi-organization federations (DAIS)," in [*24th IEEE Conference on Decision and Control*], (2017).

[11] Jain, R., Chiu, D. M., and Hawe, W., "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," Tech. Rep. DEC-TR-301, Digital Equipment Corporation (1984).

[12] Plimpton, S., "Fast parallel algorithms for short-range molecular dynamics," *Journal of Computational Physics* **117**(1), 1 – 19 (1995).

[13] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," in [*Advances in Neural Information Processing Systems 25*], Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., eds., 1097–1105, Curran Associates, Inc. (2012).

[14] Saha, S. K., Deshpande, P., Inamdar, P. P., Sheshadri, R. K., and Koustonikolas, D., "Power-thoughput tradeoffs of 802.11n/ac in smartphones," in [*2015 IEEE Conference on Computer Communications (INFOCOM)*], 100–108 (2015).

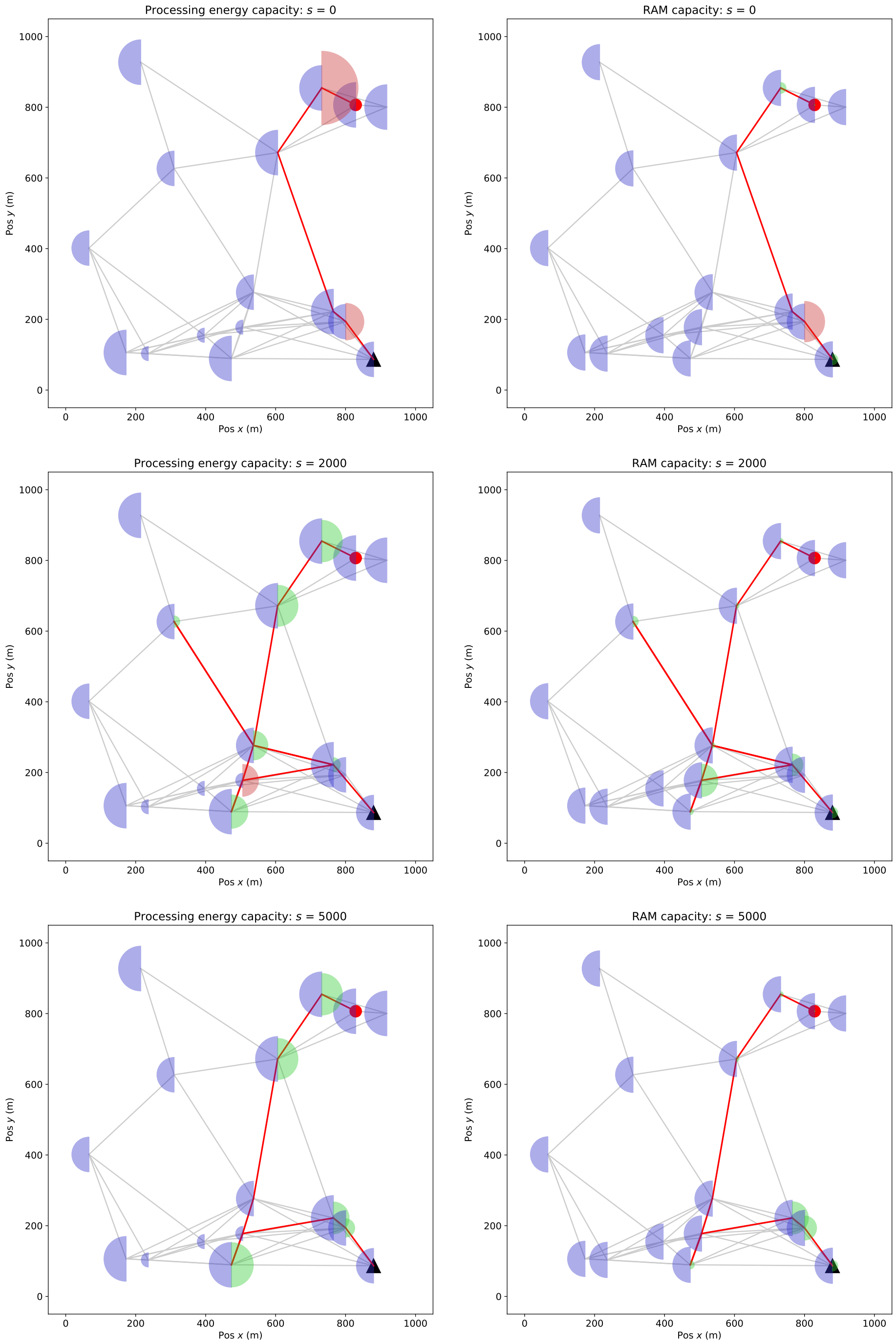

Figure 1. Experiment 1. Schematics showing the available resources on the physical graph and utilization by analytics on the logical graph for experiment 1. Blue left-semicircles represent the capacity at each physical node. Red right-semicircles represent a utilization that is over capacity, while green right-semicircles represent a utilization that is within capacity. The areas of the semicircles are proportional to the capacity/utilization, illustrating the actual utilization. The value of s denotes the iteration number.

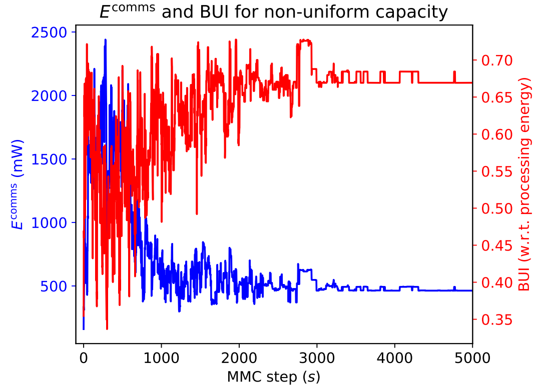

Figure 2. Experiment 1. Plot of communications energy (blue) and BUI of the processing energy (red), as a function of iteration number.

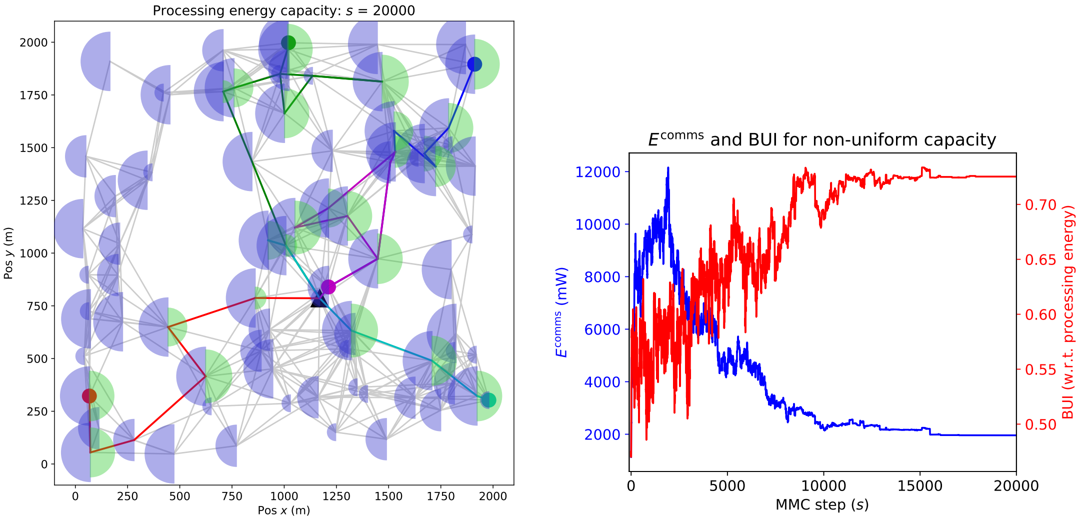

Figure 3. Experiment 2a: Left: Schematic of final processing capacity usage, embedding 5 AlexNet calculations in an 80-node physical graph, with the same method of coloring and sizing the semicircles as described for figure 1. Right: Plot of communications energy (blue) and BUI of the processing energy (red), as a function of iteration number for experiment 2a.

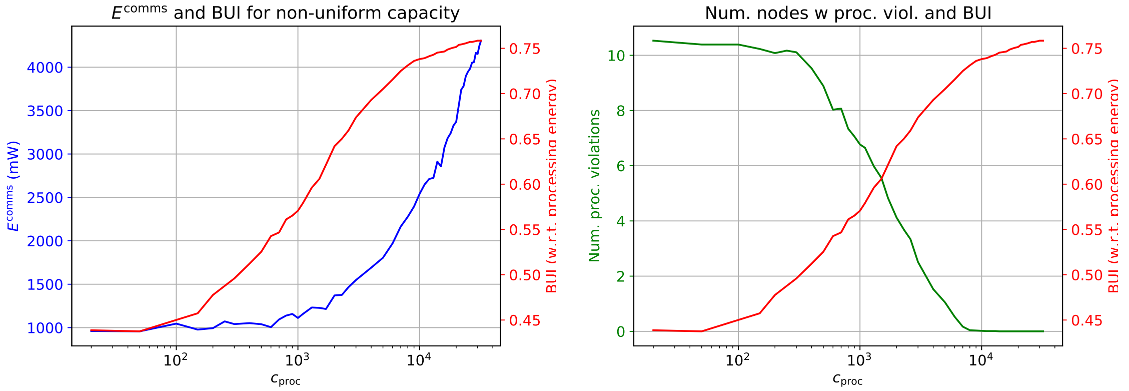

Figure 4. Experiment 2b: Plot of communications energy (blue) and BUI of the processing energy (red), as a function of iteration number.

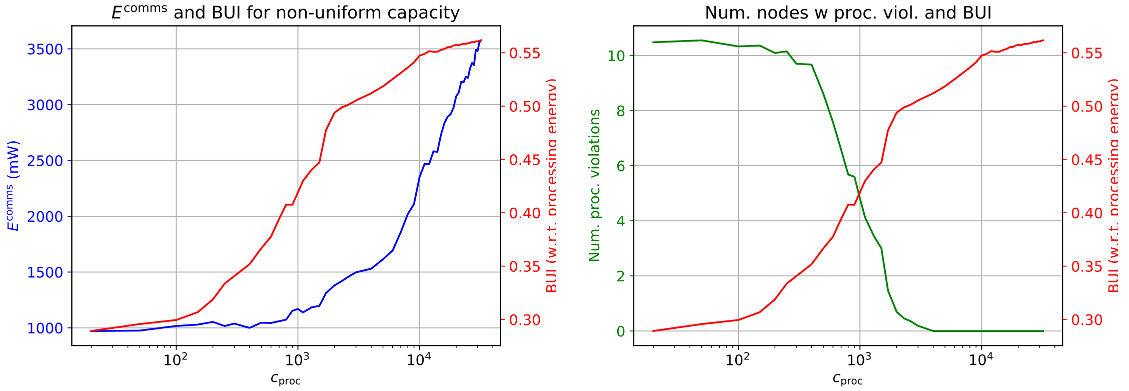

Figure 5. Experiment 3: Plot of communications energy (blue) and BUI of the processing energy (red), as a function of iteration number.