Maximal Operation Time Estimation for Modular and Self-Reconfigurable Robots with Output Current Constraints

Chi-An Chen, Thomas Collins, and Wei-Min Shen
Polymorphic Robotics Laboratory, Department of Computer Science
University of Southern California
Los Angeles, USA
e-mail: {chianc, collinst}@usc.edu, shen@isi.edu

***Abstract*— A key problem in modular and self-reconfigurable robot power sharing research is that of estimating the globally maximal operation time (MOT) of a system of connected modules that share power with one another, as this information is vital to determining resource allocation in power sharing schemes. Existing MOT estimation algorithms do not consider the effect that output current constraints at each module – i.e., the fact that each module can only share a certain amount of power with other modules – have on the estimation of MOT information. This paper proposes both centralized and distributed algorithms for estimating the MOT of systems of connected modular and/or self-reconfigurable robots in which each robot module is subject to constraints on the amount of electrical current it can output. These algorithms are based on a transformation of the power sharing problem (with output current limits) to a minimum-cost flow problem, for which efficient algorithms exist. The proposed algorithms are validated in large-scale simulations to demonstrate their correctness, feasibility, and scalability.**

***Keywords*—** *Robotic Network; Self-Reconfigurable Modular Robots; Robot; Self-Sufficiency; Energy Allocation; Energy Sharing; Energy Autonomy; Minimum Cost Flow*

I. Introduction

In recent years, there has been a growing emphasis in the modular robotics literature on optimizing the collective operation time of systems of robotic modules. A key problem in modular and self-reconfigurable robot power sharing research is that of estimating the globally maximal operation time (MOT) of a system of connected modules that share power with one another, as this information is vital to determining resource allocation in power sharing schemes.

Many modular robotic systems, notably self-reconfigurable robots (e.g., [1], [2]), are designed such that each module is itself an autonomous robot, with its own electronics, sensors, actuators, and its own independent power source. One potential advantage of such systems is that connected groups of modules could coordinate with one another to autonomously share power in a way that maximizes the operation time of the entire system. Research into modular and self-reconfigurable robot power sharing has been largely focused on designing hardware mechanisms that facilitate the transfer of power from one module to another (usually passively, without the modules having control over the amount shared, e.g. [3], [4]).

In contrast, few studies have considered the power sharing problem at an algorithmic level: how can systems of connected robot modules autonomously share power with one another in a way that maximizes the collective operation time of the entire system? In our previous paper [5], we proposed a near-optimal power sharing algorithm for self-reconfigurable modular robots that showed promising results when compared against the state-of-the-art schemes proposed in [3], [4]. One drawback of our method, however, was that it utilized the impractical assumption that each robot module could share as much current as needed with other modules in the system (provided the shared amount did not deplete the module's power resource completely). Power resources generally have output current constraints that limit the amount of outgoing current. Such constraints would cause our previously proposed method to overestimate the MOT of the system, leading to poor resource estimation and allocation.

In this paper, we propose centralized and distributed versions of a new MOT estimation algorithm which is capable of generating high-quality approximations of the MOT of a system of robot modules, even when the independent power sources of each module contain (possibly different) current output limits. This algorithm is based on a transformation of the power sharing problem with current output limits to a minimum-cost flow problem, for which efficient algorithms are known. The next section discusses necessary background information on related energy allocation approaches.

II. Background and Related Work

Energy allocation techniques have been studied in a number of domains, including smart buildings and smart grids (e.g., [6], [7], [8]), Vehicle-to-Grid research (V2G) [9], energy optimization for sensor networks [10], [11], etc.

Energy allocation problems, including those studied in the research above, are often modeled as flow problems, most notably using a minimum cost flow (MCF) problem formulation [12]. In general, the model of MCF consists of a network configuration (visualized as a graph) $G = (N, A)$, where N is a set of nodes and A is a set of arcs. Each arc includes an associated cost c_{ij} and capacity u_{ij}, where $i, j \in N$ and $(i, j) \in A$. $b(i)$ represents the supply (if $b(i) > 0$) or demand (if $b(i) < 0$) of node i. By minimizing

the objective function in Eq. 1 (below), the minimum cost flow x_{ij} on each arc (i, j) can be obtained:

$$Minimize\ z(x) = \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{1}$$

By applying MCF, we can obtain the optimal energy allocation between each node, i.e. x_{ij}, with respect to the minimization of the path cost.

In MCF problems, $b(i)$ is given as input; however, in the power sharing problem with which this paper is concerned, the supplies and demands of each robot (node) are unknown and need to be estimated online. In most related approaches (including those in [6], [8]), the algorithm must be given as input the supplies and demands of each node in order to compute a solution, making them unsuitable for solving the problem at hand.

Additionally, we note that in order to obtain a feasible solution, one of the assumptions for the input to MCF algorithms is that the total supply in the network is equal to the total demand (Eq. 2).

$$\sum_{i\in N} b(i) = 0. \tag{2}$$

Though we do not know the supplies and demands for power of each modular robot, we show that the power sharing problem with current output limits can be transformed into a minimum cost flow problem that satisfies Eq.2. In the next section, we formalize the power sharing problem for systems of modular and self-reconfigurable robots.

III. Power Sharing in Self-Reconfigurable and Modular Robot Systems

We model a modular or self-reconfigurable robot system as a graph $G = (N, A)$, where the set of graph nodes N represents the set of modular robots in the system, and the set of graph arcs A represents the connections between pairs of modules. We use $b(i)$ to denote the power supply ($b(i) > 0$) or demand ($b(i) < 0$) of each robot i. The symbol $\pi_{power\,sharing}$ denotes a power sharing policy. $rcc(i)$ and $cc(i)$ stand for remaining charge capacity (remaining units of power) and current consumption (rate of power consumption) for robot i, respectively. $T^*(G(N, A))$ represents the theoretical maximum operation time of the whole system and can be calculated as in equation (3) [5]. $\tilde{T}(i)$ represents the measured operation time of the robot i.

$$T^*(G(N, A)) = \frac{\sum_i rcc(i)}{\sum_i cc(i)},\ \forall i \in N. \tag{3}$$

Lemma 1: Without considering path loss and the regulator's efficiency, if there exists a $\pi_{power\,sharing}$ which makes each modular robot's operation time the same, then the system can reach $T^*(G(N, A))$.

Proof. After applying $\pi_{power\,sharing}$, the operation time of every robot i can be represented as equation 4, where $rcc^{\pi}(i)$ denotes $rcc(i)$ after $\pi_{power\,sharing}$ is applied. We can multiply $cc(i)$ on both sides of equation (4) and obtain equation (5).

$$\tilde{T}(i) = \frac{rcc^{\pi}(i)}{cc(i)},\ \forall i \in N \tag{4}$$

$$\Rightarrow rcc^{\pi}(i) = cc(i) \cdot \tilde{T}(i) \tag{5}$$

Based on the assumption of lemma 1, we ignore path loss and the regulator's efficiency. By summing equation (5) over all robots i, we can obtain equation (6), then derive equations (7) and (8) using (3) to establish lemma 1.

$$\sum_{i\in N} rcc(i) = (\sum_{i\in N} cc(i)) \cdot \tilde{T}(i) \tag{6}$$

$$\Rightarrow \frac{\sum_{i\in N} rcc(i)}{\sum_{i\in N} cc(i)} = \tilde{T}(i) \tag{7}$$

$$\Rightarrow T^*(G(N, A)) = \tilde{T}(i). \tag{8}$$

Lemma 2: For lemma 2, we again ignore path loss and the regulator's efficiency. If $\pi_{power\,sharing}$ can't make each modular robot's operation time be the same due to some constraints, then the system can't reach $T^*(G(N, A))$.

Proof. Starting from equation (5), since $T(i)$ is not the same for each robot i, once we sum equation (5) of each robot i, we can obtain equation (9).

$$\begin{aligned}\sum_{i\in N} rcc(i) = cc(0) \cdot \tilde{T}(0) + cc(1) \cdot \tilde{T}(1) + \\ \ldots + cc(|N| - 1) \cdot \tilde{T}(|N| - 1).\end{aligned} \tag{9}$$

$min(\tilde{T}(i))$ represents the system operation time, and it can be extracted from each $cc(i) \cdot \tilde{T}(i)$, $i \in N$ to derive equation (10).

$$\begin{aligned}&\Rightarrow \sum_{i\in N} rcc(i) = (\sum_{i\in N} cc(i)) \cdot min(\tilde{T}(i)) + \\ &cc(0) \cdot \tilde{T}(0)_{residual} + cc(1) \cdot \tilde{T}(1)_{residual} + cc(2) \cdot \\ &\tilde{T}(2)_{residual} + \ldots + cc(|N| - 1) \cdot \tilde{T}(|N| - 1)_{residual}, \\ &where\ \tilde{T}(i)_{residual} = \tilde{T}(i) - min(\tilde{T}(i)).\end{aligned} \tag{10}$$

By collecting $cc(i) \cdot \tilde{T}(i)_{residual}$ into $RCC_{unutilized}$, we can obtain equation (11) and then derive equations (12) and (13) from it.

$$\sum_{i\in N} rcc(i) = (\sum_{i\in N} cc(i)) \cdot min(\tilde{T}(i)) + RCC_{unutilized} \tag{11}$$

$$\Rightarrow \frac{\sum_{i\in N} rcc(i)}{\sum_{i\in N} cc(i)} = min(\tilde{T}(i)) + \frac{RCC_{unutilized}}{\sum_{i\in N} cc(i)} \tag{12}$$

$$\Rightarrow T^*(G(N, A)) = min(\tilde{T}(i)) + \frac{RCC_{unutilized}}{\sum_{i\in N} cc(i)} \tag{13}$$

Since both $RCC_{unutilized}$ and $\sum_{i\in N} cc(i)$ are larger than zero, $T^*(G(N, A)) > min(\tilde{T}(i))$ is valid and lemma 2 is established.

In order to maximize $min(\tilde{T}(i))$, $RCC_{unutilized}$ needs to be minimized. One primary cause of $RCC_{unutilized}$ is output current limits (i.e., limits on the number of units of

power each module can share). In the next section, we will derive how to find the optimal (maximal) system operation time even with output current constraints, a quantity that we name $\hat{T}(G(N,A))$, where $\hat{T}(G(N,A)) = max(min(\hat{T}(i)))$.

IV. Naive Solution for $\hat{T}$

In practice, the battery or power source of a robot has an output current limit. We use $B^{'}(i)$ to represent this output limit for each robot i. We use the simple example in Figure 1 to study our power sharing problem model with output current constraints.

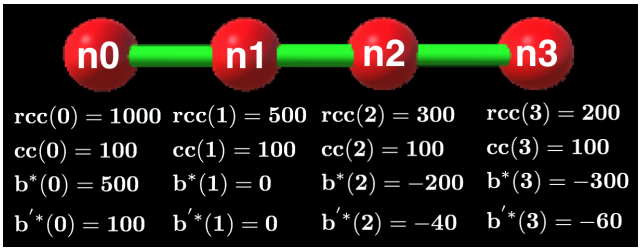


Fig. 1: A Simple Example for Illustrating Power Sharing with Output Current Constraints.

In Figure 1, $rcc(i)$ and $cc(i)$ are given, and, according to our previous research in [5], we can calculate $b^*(i)$ using Eq. (14), where $b^*(i)$ stands for the supply or demand of module i in order to reach $T^*(G(N,A))$.

$$b^*(i) = rcc(i) - cc(i) \cdot T^*(G(N,A)),\ i \in N. \tag{14}$$

Here we use $b^{*'}(i)$ (calculated in Eq. 15) to represent the average number of units of power that need to be shared per unit of time by module i in order to reach $T^*(G(N,A))$.

$$b^{*'}(i) = \frac{b^*(i)}{T^*(G(N,A))},\ i \in N. \tag{15}$$

Assume the output current constraint for each module in this system is 50 units: $B^{'}(i) = 50, \forall i \in N$. In Figure 1, since $b^{*'}(0) > B^{'}(0)$, $b^{*'}(0)$ will be clamped to $B^{'}(0)$, which will result in $n2$ and $n3$ running out of power before $n0$ donates its full amount of power. This will also cause $T(i) \neq T(j),\ \exists i,j \in N$.

Here we can make use of lemma 1 and 2 by fully utilizing $n0's$ saturated donation to reduce $RCC_{unutilized}$ and also attempt to make $T(1) = T(2) = T(3)$. This leads to the system of equations in (16):

$$\begin{cases} T(1) = T(2) \Rightarrow \dfrac{500}{100+x} = \dfrac{300}{100+y} \\ T(2) = T(3) \Rightarrow \dfrac{300}{100+y} = \dfrac{200}{100+z} \\ 50 + x + y + z = 0 \end{cases} \tag{16}$$

Solving this system of equations produces $x = 25$, $y = -25$, $z = -50$. If no current constraints existed, $n1$ would not have to donate any power; however, with the constraints, $n1$ has to act as a small donor in order to reach $\hat{T}(G(N,A))$.

It's easy to use this naive solution to solve $\hat{T}(G(N,A))$ for a small-scale system. However, as the number of nodes in the system (number of robots) increases, this algorithm quickly becomes impractical (as does acquiring $b(i)$ for each node). In self-reconfigurable and modular robotics, we are often interested in large-scale systems. Thus, it is essential to find a more practical and scalable algorithm to solve $\hat{T}(G(N,A))$.

V. $\hat{T}$ Algorithm

From the previous simple example, we see that if we rearrange the $cc(i)$, we should be able to equalize each robot's operation time except for the robot i with $b^{*'}(i) > B^{'}(i)$. This leads to the Centralized $\hat{T}$ Algorithm (Algorithm 1).

A. Centralized $\hat{T}$ Algorithm

Algorithm 1: Centralized $\hat{T}$ Algorithm

Input: $G(N,A)$, $rcc(i)$, $cc(i)$, $B'(i)$, $\forall i \in N$
Output: $\hat{T}(G(N,A))$

1 nodes ← make-priority-queue(make-node(initial-state[G(N, A)])) descending ordered by $T(i)$, $\forall i \in N$;
2 **while** $T(nodes[front]) - T(nodes[back]) > \epsilon$ **do**
3 $\quad b'(front \to back) = cc(nodes[back]) - \dfrac{rcc(nodes[back]) \cdot cc(nodes[back-1])}{rcc(nodes[back-1])}$
4 $\quad$ **if** $B'(nodes[front]) > (b'(front \to back) + 1)$ **then**
5 $\quad\quad cc(nodes[front]) \leftarrow cc(nodes[front]) + b'(front \to back) + 1$
6 $\quad\quad B'(nodes[front]) \leftarrow B'(nodes[front]) - b'(front \to back) - 1$
7 $\quad\quad cc(nodes[back]) \leftarrow cc(nodes[back]) - b'(front \to back) - 1$
8 $\quad$ **else if** $B'(nodes[front]) <= (b'(front \to back) + 1)$ **then**
9 $\quad\quad cc(nodes[front]) \leftarrow cc(nodes[front]) + B'(nodes[front])$
10 $\quad\quad B'(nodes[front]) = 0$
11 $\quad\quad cc(nodes[back]) \leftarrow cc(nodes[back]) - B'(nodes[front])$
12 $\quad\quad$ Remove-Front(nodes)
13 $\quad$ update-priority-queue(nodes)
14 return $T(nodes[front])$

In Algorithm 1, $b'(front \to back)$ stands for the number of units of power shared from $nodes[front]$ to $nodes[back]$ per unit of time and the equation in line 3 is derived from equation (17).

$$\frac{rcc(nodes[back])}{cc(nodes[back]) - b'(front \to back)} = \frac{rcc(nodes[back-1])}{cc(nodes[back-1])} \tag{17}$$

The basic idea of Algorithm 1 is to make a priority queue ordered by $T(i), i \in N$ (in descending order) and always move power from $nodes[front]$ to $nodes[back]$. Meanwhile, $B'(front)$ is monitored to see if $nodes[front]$ is saturated.

B. Distributed $\hat{T}$ Algorithm

Algorithm 2, presented below, is a distributed version of Algorithm 1. The basic idea is similar to the centralized one, but each node only knows its neighbors' information. Firstly, each node makes a priority queue named "cell" which consists of itself and its neighbors ordered (descending) by its operation time as shown in line 2. After "cell" is created, the node with the longest operation time and the positive B' value is selected to share power to the node with the shortest operation time, (i.e. cell[back]). These steps are shown in line 3 and lines 5-9. After the power sharing steps, the operation time of each node and the order in the cell are updated as shown in line 10 and 11. Lines 6-12 are repeated until the condition in line 5 is satisfied.

Although the returned value is not $\hat{T}(G(N,A))$, the final goal and the most desirable b' are obtained as shown in lines 14 and 16.

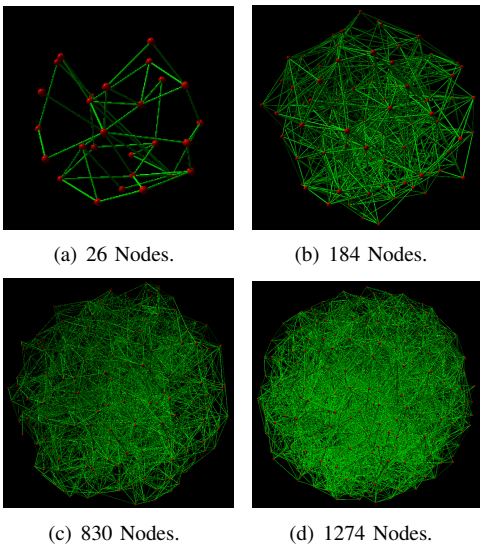
Algorithm 2: Distributed $\hat{T}$ Algorithm

1 For each node i,
  Input: $rcc(i)$, $cc(i)$, $B'(i)$, $rcc(j)$, $cc(j)$, $B'(j)$, $\forall j \in neighbor\,of\,node\,i$
  Output: $b'(i)$
2 cell $\leftarrow$ make-priority-queue(node i and node j) descending ordered by $T(k)$, $\forall k \in$ node in the cell
3 donor $\leftarrow$ find–longest-time-with-positive-B′(cell)
4 $B'(i)_{original} \leftarrow B'(i)$
5 **while** $(T(donor) - T(cell[back]) > \epsilon)$ **do**
6 $cc(donor) \leftarrow cc(donor)$ + 1 unit
7 $B'(donor) \leftarrow B'(donor)$ - 1 unit
8 $cc(cell[back]) \leftarrow cc(cell[back])$ - 1 unit
9 $B'(cell[back]) \leftarrow B'(cell[back])$ + 1 unit
10 update-operation-time(cell)
11 update-priority-queue(cell)
12 donor $\leftarrow$ find–longest-time-with-positive-B′(cell)
13 **if** $B'(i) == 0$ **then**
14 return $b'(i) = B'(i)_{original}$
15 **else**
16 return $b'(i) = B'(i)_{original} - B'(i)$

VI. Simulation and Evaluation

To verify our algorithm and evaluate its scalability, we created a number of different networks of various sizes and initialized them with random $rcc(i)$, $cc(i)$ and $B'(i)$ values for each node i. We then ran both Algorithm 1 and Algorithm 2 on them. Figure 2 shows some examples of these networks of increasing scale and complexity, while Figure 3 plots the average and standard deviation of $\hat{T}(G(N,A))$ on increasingly complex networks. Except for the nodes with saturated $B^{'}$ values, the standard deviations of $\hat{T}(G(N,A))$ for all of the test cases were very close to zero, which justifies our theories.

Finally, Figure 4 provides a visualization of the convergence of the proposed $\hat{T}$ algorithm on a configuration of 100 SuperBot modules in the ReBots self-reconfigurable robot simulator [13]. $rcc(i)$, $cc(i)$, and $B^{'}(i)$ were again randomly chosen for each node i. The color of each SuperBot module i indicates how close the operation time of that module (the current estimate of its $\hat{T}(i)$) is to the theoretical maximum operation time of the system (T^*). The green component of the color is $\frac{\hat{T}(i)}{T^*}$, while the red component of the color is $1 - \frac{\hat{T}(i)}{T^*}$. Thus, a completely red module has 0 remaining operation time, a completely green module has an operation time of T^* (or more), and all other modules have some mixture of red and green colors. As expected, at the beginning of the simulation (Figure 4 (left)), the modules had very different initial estimates of $\hat{T}(i)$. After the proposed $\hat{T}$ algorithm was performed, however (Figure 4 (right)), the estimates of $\hat{T}$ became very close together for non-saturated modules (those whose $b^{*'}(i) \leq B^{'}(i)$). Figure 4 (middle) shows an intermediate stage of the algorithm (approximately halfway through its execution). The variance of the non-saturated nodes after the algorithm converged in this example was 0.01.



(a) 26 Nodes. (b) 184 Nodes.

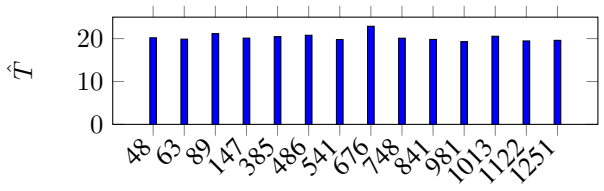
(c) 830 Nodes. (d) 1274 Nodes.

Fig. 2: Large Scale Simulations for $\hat{T}$ Algorithm.

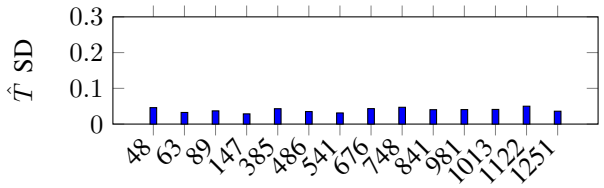
VII. Conclusion

This paper proposed both centralized and distributed algorithms to estimate $\hat{T}(G(N,A))$, the globally maximal operation time (MOT) of a system of modular or self-reconfigurable robots given a set of constraints on the amount of power that each module is able to share at any given time step. The algorithms were derived analytically and validated in large-scale numerical simulations.

Deriving an algorithm to obtain $\hat{T}(G(N,A))$ is an important milestone, as it provides a method to transform the network from one with unbalanced supplies and demands into a network whose supplies and demands meet the balance conditions necessary to utilize minimum cost flow algorithms. This also provides an opportunity to apply wireless power optimization [14].



(a) $\hat{T}(G(N, A))$ Average for Various Scale Networks



(b) $\hat{T}(G(N, A))$ Standard Deviation for Various Scale Networks

Fig. 3: Power Sharing Simulation Numerical Results for Caterpillar Configuration.

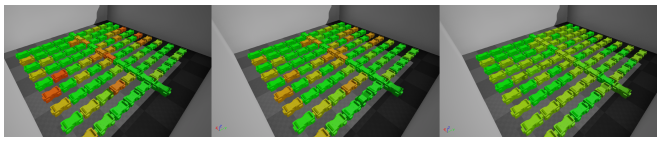


Fig. 4: A visualization of the $\hat{T}$ algorithms proposed in this paper.

VIII. Future Work

The $\hat{T}$ algorithm provides a good estimate of the resource budget of each node in the network. However, as the network size increases, the distance between each robot (node) becomes an increasingly important issue for resource planning: the longer the path between two nodes sharing power, the more power will potentially be lost during sharing. In future work, we will incorporate path loss into our model and attempt to determine optimal power sharing routes between nodes (e.g., by using minimum cost flow algorithms to find minimal cost routes).

References

[1] W.-M. Shen, H. Chiu, M. Rubenstein, and B. Salemi, "Rolling and climbing by the multifunctional superbot reconfigurable robotic system," in *Space Technology and Applications International Forum-STAIF 2008*, Melville, NY, Feb. 2008, pp. 839–848.

[2] M. W. Jorgensen, E. H. Ostergaard, and H. H. Lund, "Modular atron: Modules for a self-reconfigurable robot," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 2. IEEE, 2004, pp. 2068–2073.

[3] H. Qadir Raja and O. Scholz, "Dynamic Power Distribution and Energy Management in a Reconfigurable Multi-Robotic Organism," *ArXiv e-prints*, July 2012.

[4] R. H. Qadir, *Self-Sufficiency of an Autonomous Reconfigurable Modular Robotic Organism*. Springer Publishing Company, Incorporated, 2014.

[5] C.-A. Chen, T. Collins, and W.-M. Shen, "A near-optimal dynamic power sharing scheme for self-reconfigurable modular robots," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, May 2016.

[6] P. H. Nguyen, W. L. Kling, G. Georgiadis, M. Papatriantafilou, L. A. Tuan, and L. Bertling, "Distributed routing algorithms to manage power flow in agent-based active distribution network," in *2010 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT Europe)*, Oct 2010, pp. 1–7.

[7] P. H. Nguyen, W. L. Kling, and J. M. A. Myrzik, "Power flow management in active networks," in *2009 IEEE Bucharest PowerTech*, June 2009, pp. 1–6.

[8] B. Kim and O. Lavrova, "Optimal power flow and energy-sharing among multi-agent smart buildings in the smart grid," in *2013 IEEE Energytech*, May 2013, pp. 1–5.

[9] P. Yi, Y. Tang, Y. Hong, Y. Shen, T. Zhu, Q. Zhang, and M. M. Begovic, "Renewable energy transmission through multiple routes in a mobile electrical grid," in *ISGT 2014*, Feb 2014, pp. 1–5.

[10] S. Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh, "Optimal energy aware clustering in sensor networks," *Sensors*, vol. 2, no. 7, pp. 258–269, 2002, copyright - Copyright MDPI AG 2002; Last updated - 2014-06-20. [Online]. Available: http://libproxy.usc.edu/login?url=http://search.proquest.com.libproxy1.usc.edu/docview/1537597712?accountid=14749

[11] F. Ye, A. Chen, S. Lu, and L. Zhang, "A scalable solution to minimum cost forwarding in large sensor networks," in *Proceedings Tenth International Conference on Computer Communications and Networks (Cat. No.01EX495)*, 2001, pp. 304–309.

[12] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[13] T. Collins and W.-M. Shen, "Rebots: A drag-and-drop high-performance simulator for modular and self-reconfigurable robots," in *ISI Technical Reports*, November 2016.

[14] L. Xie, Y. Shi, Y. T. Hou, and A. Lou, "Wireless power transfer and applications to sensor networks," *IEEE Wireless Communications*, vol. 20, no. 4, pp. 140–145, August 2013.