Simulated Annealing for Optimal Resource Allocation in Wireless Networks with Imperfect Communications

Jaewook Kwak

Ness B. Shroff

Abstract—Simulated annealing (SA) method has had significant recent success in designing distributed control algorithms for wireless networks. These SA based techniques formed the basis of new CSMA algorithms and gave rise to the development of numerous variants to achieve the best system performance accommodating different communication technologies and more realistic system conditions. However, these algorithms do not readily extend to networks with noisy environments, as unreliable communication prevents them from gathering the necessary system state information needed to execute the algorithm. In recognition of this challenge, we propose a new SA algorithm that is designed to work more robustly in networks with communications that experience frequent message drops. The main idea of the proposed algorithm is a novel coupling technique that takes into account the external randomness of message passing failure events as a part of probabilistic uncertainty inherent in stochastic acceptance criterion of SA. As a result, the algorithm can be executed even with partial observation of system states, which was not possible under the traditional SA approach. We show that the newly proposed algorithm finds the optimal solution almost surely under the standard annealing framework while offering significant performance benefits in terms of its computational speed in the presence of frequent message drops.

I. INTRODUCTION

In modern wireless network systems, many network functionalities involve solving complex network-wide decision problems. Example network problems include media access control, routing optimization, resource allocation, and QoS provisioning in wireless networks, etc. A common goal pursued in these problems is to achieve the desired performance objective by seeking the best configuration of a set of system parameters. This requirement naturally leads to form a certain combinatorial optimization problem to be solved in distributed settings. However, these problems are often very difficult and high-dimensional such that their complexity grows rapidly with the size of the network.

In this paper, we consider an important class of optimization problems that are primarily motivated by resource allocation and link scheduling problems in wireless networks. A classical example of such problems is the max-weight or weighted sum rate maximization problems, which serves as a basis for many resource management and network design problems.

These problems are typically difficult to solve, and are in general known to be NP-hard even in the simple binary capacity model. In addition, emerging wireless communication technologies employ increasingly complex adaptive modulation and coding techniques, which further exacerbate the complexity of these problems. We focus on a class of NP-hard type resource allocation problems which are often intractable to solve in an efficient way and even in a centralized manner.

The solution methodology we develop in this paper is based on the classical Simulated Annealing (SA) method [12], which is a randomized technique for approximating the optimum for a given objective function. The algorithmic procedure of SA is intuitive and simple. In each step, a trial state is randomly generated and its performance objective is evaluated. If the trial state improves the objective, the current state is replaced by the new state. If the objective of the trial state is not better than that of current one, the trial is accepted or rejected based on a certain probabilistic criterion. The advantages of SA are the relative ease of implementation and the ability to provide good solutions with provable guarantees for any arbitrary systems and objective functions. Since SA is such a ubiquitous method, it has found wide-spread applications in various engineering problems [18], [24], [9].

An integral step needed to realize SA in practical systems is the correct evaluation of the performance objective on each system state, or at least the performance differential between the current and trial states. In a distributed network where the performance objective is dependent on multiple system variables across different nodes, the task of measuring the objective differentials can be done by implementing a proper message passing mechanism. For wireless resource allocation problems, to which SA is applied, most works implicitly assume that these message exchanges are perfect. However, since wireless communication is inherently unreliable (e.g., due to fading and interference, etc.), the message transmissions containing the information about evaluating the objective may not always be successful, resulting in failure of acquisition of the information at the intended time of operating the algorithm. Our numerical evaluation reveals that a straightforward solution using SA to circumvent this problem performs very poorly in terms of its computational speed, and thus appears to be far from being practical in a situation where the message drop rate is high. The main purpose of this paper is to develop an efficient way of implementing the SA algorithm

for wireless networks even under a physical channel that experiences frequent message drops.

The main contributions of this paper are as follows.

1) We investigate an important performance issue that arises from the unreliable nature of wireless communications in implementing the SA algorithm in general distributed wireless networks.

2) We propose a new algorithmic approach that can deal more efficiently with an impact from the imperfect communications, and rigorously prove the *optimality* of the proposed algorithm under the standard SA framework.

3) We demonstrate that the proposed algorithm offers significant improvement in terms of its computational speed in networks with high message drop rates.

We organize this paper as follows. First, we provide a brief overview of related work in Section II, and some preliminaries in Section III. In Section IV, we describe the detailed implementation structure of an algorithm that is based on the SA approach, and describe the main problem we focus on in this paper. In Section V, we present our new idea to deal with the problem, along with a mathematical analysis for the optimality and efficiency of our solution. Section VI provides numerical evaluations that support our main arguments. In Section VII, we discuss some practical considerations and conclude the paper in Section VIII.

II. RELATED WORK

The scope of this paper is closely related to the problem of designing wireless link scheduling algorithms. In particular, the message issue we have introduced in the previous section is of importance in the development of recently studied CSMA-type distributed algorithms [17], [11], [22]. We provide a brief overview of previous works, and emphasize again the significance of our contributions in this context.

Recently, a suite of CSMA-type algorithms have gained a lot of attension in the research community. These algorithms are known to be throughput optimal and can be easily implemented in a distributed manner requiring minimal message overheads. The key enabler of this success is the utilization of an SA-like algorithm to solve the max-weight problem. While the goal of achieving throughput optimality is to generate a sequence of schedules such that the long-term service rates can support any feasible arrival rates, the task of solving max-weight problem plays a critical role in this job and it can be indeed leveraged to achieve optimality.

We should point out that the type of messaging used in these algorithms depends on which capacity model is used in their problem setting. In earlier works [22], [8], [11], [17], the algorithms are typically developed under a simple binary capacity model: each link can be either active or inactive, where activation of two links at a close distance leads to collision, i.e., both transmissions fail. In this model, there is few restrictions on the way in which the necessary information is collected. This is because the only information needed to decide whether to activate a link's transmission is to know whether any of its neighboring links (the set of links

that interfere with it) is active. This can be easily done by having each active neighboring link send a one-bit signal on a predefined and commonly shared time slot in order to convey its activity state, and the link simply detects the presence of the combined signal.

However, this information acquisition scheme may not be able to be used on other more realistic capacity models. One such an example is the Signal-to-Noise Ratio (SINR) model, in which links obtain capacity proportional to the ratio of their signal strength to the interference experienced in their receiver. The reason is that in this model there may not exist a clear condition that distinguishes between collision and not collision, but the degree of capacity degradation caused by the activation of other links may be different for different links depending on their transmission power mode, geographical distance between them, and etc. In this case, more detailed information about the capacity degradation from different nodes may have to be collected individually. Indeed, this increased message complexity can be a critical source of the incomplete message acquisition problem as we will explain later.

There are a few works that have extended the CSMA algorithms to the SINR model case [23], [20]. However, these works are restricted to the use of a *threshold-type* capacity model, i.e., a link obtains a unit capacity if its SINR is above a certain threshold, and zero otherwise. This condition is a critical assumption that allows to use the above mentioned information acquisition scheme and avoids the message complexity problem. This capacity model, however, does not allow the wireless nodes to use adaptive modulation and coding techniques to increase data rates for higher SINR.

In [2] and [21], the authors have considered general capacity models, not restricted to threshold-type ones. However, they ignore the message complexity issue, and assumed that all suitably defined local information needed to perform their algorithms is readily available at the time of operating the algorithm. In this paper, we do not assume such an oracle, but explicitly consider the impact due to imperfect collection of required information, and develop a solution to the problem.

It is also worthwhile to mention that the delay performance of these algorithms in queueing systems is highly affected by their computational speed. According to the standard queueing theory [7], [1], the correlation on arrival and/or service processes has an adverse impact on the queueing delay. As we will show later, our new solution is very helpful in improving the algorithm operation speed, which in turn generates more rapidly evolving and less correlated link service processes in comparison to a naive approach. From this view point, the significance of the aforementioned contributions in the scheduling problem can be translated into the fact that our proposed algorithm, applied to max-weight type problems with any general capacity model, guarantees throughput-optimality while reducing delay performance degradation due to the im-

---

Some overheads such as headers and/or guard times may be necessary depending on the types of practical systems, as in [17].

perfect communications. However, aside from this significant merit, achieving faster computational speed to generate an equivalent solution is evidently desirable in designing this type of randomized algorithms for many applications.

III. PRELIMINARIES

A. System model and objective

Network model. We consider a wireless network consisting of a set $\mathcal{N}$ of n communication links (transmitter-receiver pairs). Each link-i transmitter node has its local parameter x_i that determines its transmission power level from a discrete set $\{0, \ldots, P^{\max}\} \triangleq \mathcal{M}$. Let $\mathbf{x} = \{x_1, \ldots, x_n\}$. Links interfere with each other such that a transmission of one link is treated as interference at other links. We consider the SINR-based interference model. That is, when each link i sends a signal with its power level x_i, the receiver of each link i attains its SINR level, $\gamma_i(\mathbf{x}) = \frac{g_{ii} x_i}{\sum_{j \neq i} g_{ji} x_j + n_0}$, where g_{ij} is the channel gain from link-i transmitter to link-j receiver, and n_0 is the thermal noise. The link i then obtains its transmission rate $c_i(\gamma_i(\mathbf{x}))$ as a function of the experienced SINR level, which is a typically monotone function, such as $\log(1 + \gamma_i(\mathbf{x}))$. Let $\mathcal{X} \triangleq \mathcal{M}^n$, and call an instance $\mathbf{x} = \{x_1, \ldots, x_n\} \in \mathcal{X}$ *configuration*. We denote by $c(\mathbf{x}) = \{c_i(\mathbf{x})\}_{i \in \mathcal{N}}$ a capacity vector with configuration $\mathbf{x}$. We will also use $\mathbf{x}_{[S]} = \{x_i\}_{i \in S}$, for $S \subseteq \mathcal{N}$ to denote a subset S of configuration $\mathbf{x}$.

Main objective. We require that each link controls its transmission power level in order to achieve a certain performance objective. Specifically, we aim at designing a distributed algorithm that makes decisions $\mathbf{x}(t) \in \mathcal{X}$ so that the long-term time proportion of the configuration converges to a solution to

$$\text{(OPT-MW)} \quad \text{maximize}_{\mathbf{x} \in \mathcal{X}} \quad \sum_{i \in \mathcal{N}} w_i c_i(\mathbf{x}),$$

where w_i is a weight of link i. In the following, we call the pair of product $w_i c_i(\mathbf{x}) \triangleq f_i(\mathbf{x})$ *performance objective*, $f_i : \mathcal{X} \to \mathbb{R}$, of link i associated with each configuration $\mathbf{x} \in \mathcal{X}$.

A significant motivation for considering **OPT-MW** is its relevance to the throughput-optimality in queueing systems. To be more specific, suppose that each link maintains a queue fed by an exogenous packet arrival process. In [16], it was shown that if in each time slot, a configuration is selected according to the above max-weight rule, where the weight is queue size, then the queues can be stabilized (keeping all link queues finite) for all arrival vectors that are within the capacity region determined by the convex combination of capacity vectors with all possible configurations. While in our problem setting the weight parameters are assumed to be constant, an algorithm that solves **OPT-MW** with large enough weights can be shown to be throughput-optimal based on the time-scale separation assumption, and the assumption can be relaxed by the recent queue-based adaptation schemes [22], [8].

In general, **OPT-MW** is known to be an NP-hard problem, and therefore it is unlikely that there exists an efficient algorithm to solve it even in a centralized manner. Our solution approach to the problem is to utilize the *simulated annealing* method, which is known to guarantee to find the optimal

solution with high probability in a certain asymptotic sense even for NP-hard problems.

B. Simulated annealing

Central to the idea of simulated annealing is the Metropolis Hastings (MH) algorithm, which is a Monte Calro Markov Chain (MCMC) method that can be used for obtaining a sequence of samples from a given probability distribution. We here briefly review the MH algorithm and its relation to simulated annealing to solve **OPT-MW**.

Consider an irreducible Markov chain X_t with a finite state space Ω and its transition probability matrix $\mathbf{P} = \{P_{ij}\}_{i,j \in \Omega}$. Let $\boldsymbol{\pi} = \{\pi\}_{i \in \Omega}$ be a probability distribution over the state space. The MH algorithm is intended to obtain a transition probability matrix $\mathbf{P}$ that has π as its stationary distribution while satisfying the reversibility condition, i.e., $\pi_i P_{ij} = \pi_j P_{ji}$. The details of the MH algorithm are described as follows. At the current state i of X_t, the next state X_{t+1} is proposed with a probability with *proposal* distribution c_{ij} - the state transition probability of an arbitrary irreducible Markov chain on the same state space, where $c_{ij} > 0$ if and only if $c_{ji} > 0$. The proposed state transition is accepted with probability $\alpha_{ij} = \min\left\{1, \frac{\pi_j c_{ji}}{\pi_i c_{ij}}\right\}$, and is rejected with probability $1 - \alpha_{ij}$. Therefore, the transition probability P_{ij} is given by $P_{ij} = c_{ij} \alpha_{ij} = \min\{c_{ij}, c_{ji} \pi_j / \pi_i\}$, for $i \neq j$, and $P_{ii} = 1 - \sum_{j \neq i} P_{ij}$. When the proposal distribution is symmetric, i.e., $c_{ij} = c_{ji}$ for all $i, j \in \Omega$, the form of transition probabilities reduces to $P_{ij} = c_{ij} \min\{1, \pi_j / \pi_i\}$.

As an application of the MH algorithm, an important class of probability distribution to be used for solving combinatorial optimization problems is *Boltzmann-Gibbs distribution*, which is typically constructed for **OPT-MW** by

$$\pi(\mathbf{x}) = \frac{1}{Z} e^{\beta f(\mathbf{x})}, \quad \mathbf{x} \in \mathcal{X}, \tag{1}$$

where $f(\mathbf{x}) = \sum_{i \in \mathcal{N}} f_i(\mathbf{x})$, Z is the normalization constant: $Z = \sum_{\mathbf{x}' \in \mathcal{X}} e^{\beta f(\mathbf{x}')}$, and $\beta > 0$ is a parameter related to capturing the trade-off between optimality and convergence speed. Evidently, as β becomes large, the probability distribution will be concentrated on the set of optimal solutions $\mathcal{X}^* := \{\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) = \max_{\mathbf{x}' \in \mathcal{X}} f(\mathbf{x}')\}$. In this form of $\boldsymbol{\pi} = \{\pi(\mathbf{x})\}_{\mathbf{x} \in \mathcal{X}}$, the constructed transition probabilities by the MH algorithm can be written as $P(\mathbf{x}, \mathbf{x}') = c(\mathbf{x}, \mathbf{x}') e^{-\beta[f(\mathbf{x}) - f(\mathbf{x}')]^+}$, for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$) and $P(\mathbf{x}, \mathbf{x}) = 1 - \sum_{\mathbf{x}' \in \mathcal{X}} P(\mathbf{x}, \mathbf{x}')$, given that a suitably defined proposal distribution $c(\mathbf{x}, \mathbf{x}')$ is symmetric.

The simulated annealing is an adapted version of the MH algorithm. The most distinct feature of SA is that it allows β to increase monotonically in time, but with sufficiently slowly varying rate, in order to guarantee the convergence to the optimal solution in a certain probabilistic sense. The time-varying parameter $T(t) = 1/\beta(t)$ is often referred to as the *temperature* at time t, and the sequence of $T(t)$ is called *cooling schedule*. Many proofs of convergence of cooling schedules have already appeared in the literature [4], [10]. We defer discussion of this topic in Section V-B.

Algorithm 1 Basic SA (BSA) Algorithm (in time slot t)

Pick phase:
 1: The network selects a link $i \in \mathcal{N}$ u.a.r.
 2: The link i chooses $x_i(t) \in \mathcal{M} \backslash \{x_i(t-1)\}$ u.a.r.
 3: Set $x_j(t) = x_j(t-1), \forall j \in \mathcal{N} \backslash \{i\}$.

Train phase:
 4: Test the new configuration $\mathbf{x}(t)$.
 5: Every link $j \in N$ locally measures $f_j(\mathbf{x}(t))$.
 6: Set $\Delta_j = f_j(\mathbf{x}(t)) - f_j(\mathbf{x}(t-1))$.

Messaging phase:
 7: Each link $j \in N_i$ sends Δ_j to link i.

Decision phase: (at node i)
 8: Set $\Delta = \Delta_i + \sum_{j \in N_i} \Delta_j$.
 9: **if** $\Delta \leq 0$ **then** $x_i(t) = x_i(t-1)$ w.p. $1 - e^{\beta \Delta}$

IV. THE IMPLEMENTATION AND THE CHALLENGE

A. Implementation structure

Realizing the SA idea in a distributed network requires a considerable attention since the specific implementation in practical networks will differ greatly depending on the characteristics and the constraints of the network systems. We present our implementation structure of the SA idea to be performed in the SINR model.

In our implementation, time is divided into discrete time slots where each time slot t consists of four phases which include pick, training, messaging, and decision. In the pick phase, the network selects a link $i \in \mathcal{N}$ uniformly at random (u.a.r.). The task of selecting a random link can be done in a distributed manner by having each link trigger an independent poisson clock with a unit rate over continuous time domain, and by suitably defining a time slot as an interval of each clock tick. The selected link generates a new power level state $x_i(t)$ u.a.r. different from its previous state $x_i(t-1)$. The newly generated configuration $\mathbf{x}(t)$ is then tested by having each transmitter node transmit a test signal with the selected power level, and the receiver node of each link measures its performance objective. Each receiver node then constructs a message containing the objective differential - the measured quantity subtracted by that of previous time slot - and transmits it to the transmitter node of link i during the messaging phase. Upon receiving the messages, the link i decides whether to accept the new power level state or not, based on the received information and the previously described MH algorithm to achieve $\boldsymbol{\pi}$ in Eq (1).

Note that if links are located sparsely over a geographical region and the channel gain quickly decreases with the distance between a receiver and an interfering transmitter, then it is reasonable to assume that the interference from links that are far away can be ignored. Specifically, we define a neighbor set N_i for each link i such that a link j belongs to the neighbor set N_i if link-j receiver is located within a given radius of link-i transmitter, and will consider only those links in the neighbor set as the primary sources of the interference. Therefore, each link-i transmitter node only needs to collect information from

its neighboring links $j \in N_i$ during the messaging phase. The detailed algorithm is outlined in Algorithm 1.

At first glance, our implementation appears to be similar to the standard PICK-and-COMPARE methods as introduced in [6], [13], [15]. The main idea of the previous approaches is to have *every* node generate its new random power level, and compare its objective value with that of the previous power allocation. If the new power allocation improves the objective value, then the new allocation is accepted to use in the next time slot, and if otherwise, remains to use the previous one. However, in multi-hop wireless networks, this comparison task is very challenging because it requires to compare the network-wide weighted-sum rates achieved by the two power allocation. To this end, they adopted a gossip-like algorithm, however, the computation of each power allocation using the gossip algorithm requires up to $O(n^3)$ information exchange, which may not be easily implementable for large networks. On the other hand, we do not require such a network scale comparison, as we perform the comparison task at link level. That is, we propose to change only a single state at a time, which makes the computation of the objective differential easy and suitable to be implementable in a distributed manner.

B. The challenge with imperfect communications

We have described the basic SA algorithm based on the assumption that the message containing the objective differential locally measured at each node is perfectly delivered to the intended node during the messaging phase. In practice, however, the delivery of messages may not always be successful, and there can be several reasons that can prevent the message delivery from being successful.

1) *Fading.* A primary reason for the delivery failure is due to the inherently unreliable nature of wireless communications. In wireless communications, the transmission channel suffers from temporal variations in its condition with various variables, and this can often result in a great amount of signal attenuation and message decoding errors.

2) *Message complexity.* When the network experiences frequent events of join and leave of nodes, it may not be easy for each node to find a proper coordination in a deterministic way for receiving multiple messages from different neighbors. To deal with such a potential dynamics, an *Aloha*-type of randomized neighbor discovery method can be used as an alternative, e.g. [25]. One way to do is to allocate multiple sub-slots during the messaging phase, and in each sub-slot, nodes transmit their message with some probability. In this way, nodes can deliver messages while avoiding collisions in a randomized fashion. However, there is always a chance that the delivery of messages may not be successful, since the number of sub-slots is finite and fixed.

We capture various factors that can cause message drops by means of probability to represent the combined effect. In specific, we assume that in each time t, the selected node i is only able to collect a subset $S(t)$ of nodes from its neighbors N_i with some unknown probability $q_{i,S(t)}$, $S(t) \subseteq N_i$, which

is $i.i.d.$ over time slots, where the probability of collecting the full set information is assumed to be non-zero, i.e., $q_{i,N_i} > 0$.

This limited capability of the message passing poses the following practical challenge: when the subset $S(t)$ of information collected at time t is strictly smaller than N_i, the node i cannot compute the state transition probability correctly, and therefore it is unclear how to behave in this time slot. A straightforward idea to deal with the problem is as follows. If the intended node gathers all the information from the full set of its neighbors successfully, then the node performs Algorithm 1. And, if otherwise, it defers performing the algorithm and simply maintains the current state (Algorithm 2).

Algorithm 2 Lazy SA (LSA) algorithm (at node i in time t)
___
Message Input : $S(t) \subseteq N_i$ and $\{\Delta_j\}_{j \in S(t)}$.
Decision phase:
  1: **if** $S(t) \equiv N_i$ **then** perform (8-9) in Algorithm 1
  2: **else** $x_i(t) = x_i(t-1)$
___

We verify that this algorithm has its stationary distribution as π in Eq. (1), of which proof is in Appendix.

Proposition 1: The stationary distribution of LSA algorithm is π.

As one can notice, the problem of this algorithm is its slow computational speed. Suppose that a node has multiple neighbors and the messages each from different neighbors drop independently with some non-zero probability. Then, the probability that it obtains all the information so that it can perform the algorithm decreases exponentially fast with the number of neighboring nodes. Next, we present a new approach that can greatly improve the algorithm operation speed in the presence of message drops.

V. IMPROVING THE COMPUTATIONAL SPEED

A. Proposed solution: rapid SA (RSA) algorithm

The high level description of the main idea we introduce here is as follows. In many network application scenarios, a change of a single nodal configuration often results in a limited amount of impact to the dependent performance objectives. With the knowledge of the bounded impact, we construct a confidence range on the objective differential that can be made due to the change of configuration, and utilize it to compute the desired level of probabilistic uncertainty in the stochastic acceptance criterion of SA. As a result, a certain level of impreciseness on the evaluation of objective differentials can be tolerated without affecting its optimality.

To give a motivating example, consider the following simple network scenario with a set of four nodes, $\mathcal{N} = \{a, b, c, d\}$, where each node represents a distinct pair of communication link. Each node $i \in \mathcal{N}$ can be either active, $x_i = 1$, or inactive, $x_i = 0$, and two nodes connected in the graph (presented in Fig 1) *conflict* with each other such that a node obtains a unit capacity only if it is active and all its neighbor nodes (the set of nodes connected to it in the graph) are inactive,

and obtains zero capacity if otherwise. The objective is to maximize $f(\mathbf{x}) := \sum_{i \in \mathcal{N}} w_i c_i(\mathbf{x})$ where the weights w_i's are chosen as $w_a = 5, w_b = 7, w_c = 10, w_d = 3$. With this setup, suppose that the configuration at current time t is $\mathbf{x}(t) = \{1, 1, 0, 0\}$, i.e., only a and b are active, and consider to switch the state of node c from inactive to active, i.e., $\mathbf{x}(t+1) = \{1, 1, 1, 0\}$ according to the SA framework. In this case, the local objective differential measured (during the train phase) at each node is $\Delta_a = -5$, $\Delta_b = -7$, $\Delta_c = 0$, and $\Delta_d = 0$, respectively, and the values $\Delta_a, \Delta_b, \Delta_d$ are to be transmitted in the messaging phase towards node c. Suppose further in this particular time slot, Δ_a and Δ_b are delivered successfully whereas Δ_d has not reached node c due to a temporally bad condition experienced over the communication channel. Since node c did not receive Δ_d, it cannot correctly compute the aggregate objective differential which is needed to compute the transition probability. On the other hand, with the knowledge of Δ_a and Δ_b, node c can determine a bounded range on the consequential aggregate objective differential such that $\Delta_a + \Delta_b + \Delta_c + \Delta_d = \Delta \in [-15, -12]$, since $\Delta_d \in \{-3, 0\}$ can be easily inferred by node c: $\Delta_d = -3$ if node d was active, and $\Delta_d = 0$ if it was inactive. Our main idea we propose here is to suggest to make a transition based on the lower bounded transition probability ($e^{-15\beta}$ in this case, rather than $e^{-12\beta}$, that with the true objective differential) that can be computed based on any subset information.

This new idea relies on the following assumption: each node i has a known lower bound (upper bound in minimization problem) on the differential contribution to the objective f_j of any neighboring node j that can be made due to solitary change of node i's configuration from x_i to x_i' such that

$$\min_{\mathbf{x}_{[-i]}} f_j(x_i', \mathbf{x}_{[-i]}) - \max_{\mathbf{x}_{[-i]}} f_j(x_i, \mathbf{x}_{[-i]}) \geq b_{x_i,x_i'}^{ij},$$

where $\mathbf{x}_{[-i]} = \mathbf{x}_{[\mathcal{N} \setminus \{i\}]}$, and it is allowed to have $b_{x_i,x_i'}^{ij} = -\infty$ in the case that there is no known bound for it. For the max-weight problem under the SINR model, one can obtain a trivial bound: $b_{x_i,x_i'}^{ij}$ is $-w_j c_j^{\max}$ if $x_i' \geq x_i$ and is zero if otherwise, where $c_j^{\max}$ is the (a priori known) maximum achievable rate of link j due to physical constraints of wireless technology in use, and w_j can be easily informed as it only requires a one-time transmission. It is possible to obtain a tighter bound if additional information on the objective function, such as the gain term g_{ij} between node i and j, is available. The efficiency of this approach essentially relies on the tightness of the bounds, however, we observe through extensive simulations that loose bounds are often sufficient to offer substantial improvement on the algorithm operation speed when the packet drop rate is high. A formal description of this idea is presented in Algorithm 3.

In RSA algorithm, nodes are allowed to perform the algorithm based on the bounded estimate on the potential objective differential, which can be computed based on the subset information currently observed. Compared to the LSA algorithm, we add additional transitions on the system dynamics, so its faster computational speed is expected. In the
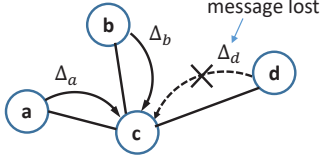
Fig. 1. An example topology in which the messages Δ_a and Δ_b are delivered successfully to node c, whereas the massage Δ_d is lost.

Algorithm 3 Rapid SA (RSA) algorithm (at node i in time t)

Message Input : $S(t) \subseteq N_i$ and $\{\Delta_j\}_{j \in S(t)}$.
Decision phase:
1: Set $\Delta_{[S(t)]} = \Delta_i + \sum_{j \in S(t)} \Delta_j + \sum_{j \in \mathcal{N}_i \setminus S(t)} b^{ij}_{x_i(t-1), x_i(t)}$.
2: **if** $\Delta_{[S(t)]} \leq 0$ **then** $x_i(t) = x_i(t-1)$ w.p. $1 - e^{\beta \Delta_{[S(t)]}}$

---

previous case, for example, when the message Δ_d was lost, the network configuration had to remain on the same state in LSA algorithm, whereas now it has some degree of probability that can transit to a new state in RSA algorithm. We obtain the following relation among the algorithms, of which proof is provided in Appendix.

Proposition 2: Let $\mathbf{P}^B$, $\mathbf{P}^R$, and $\mathbf{P}^L$ denote the transition probability matrices of BSA, RSA, and LSA algorithms, respectively. Then, for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$), it holds

$$P^B(\mathbf{x}, \mathbf{x}') \geq P^R(\mathbf{x}, \mathbf{x}') \geq P^L(\mathbf{x}, \mathbf{x}').$$

B. Optimality

Note that RSA algorithm does not necessarily achieve the same stationary distribution $\boldsymbol{\pi}$ we intended, and it is difficult to find a closed form solution for it. Technically speaking, the algorithm experiences *bias* on the desired stationary distribution due to the additional transitions we added onto the algorithm. For this reason, the conceptual argument that the probability distribution gets concentrated on the optimal states as β grows cannot be used. Our main concern here is therefore to see if the algorithm is still able to find optimal solutions under the standard SA framework. To that end, we first formally define the notion of an algorithm being optimal.

Definition 1: An algorithm is called *annealing-optimal* if a Markov chain, $X(t)$, governed by the algorithm with a proper cooling schedule for $\beta(t)$ achieves

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{P}\{X(t) \in \mathcal{X}^*\} = 1. \tag{2}$$

In the conventional SA, the cooling schedule is typically constructed by $\beta(t) = \log(t)/d$ where d is some positive constant that determines the order of cooling rate. Using this cooling schedule, by the proper cooling schedule we mean that an algorithm is said to be annealing optimal if Eq. (2) can be verified for sufficiently large enough d.

To verify the optimality of RSA algorithm, we adopt a technical method introduced in [4], in which the optimality of the original simulated annealing algorithm is proven. The authors in [4] have verified the annealing optimal of SA algorithms for

a certain class of Markov chains whose transition probabilities can be written as

$$p_{ij}(t) = c_{ij}\epsilon(t)^{V_{ij}}, \tag{3}$$

where $V_{ij}, c_{ij} \geq 0$, for all i, j, $\sum_{j \neq i} c_{ij} = 1$, for all i, $p_{ii}(t) = 1 - \sum_{j \neq i} p_{ij}(t)$, and $0 \leq \epsilon(t) \leq 1$, $t \geq 1$ is the parameter related to the cooling schedule. Note that the conventional simulated annealing algorithm can be represented by this form with setting $V_{ij} = [f(j) - f(i)]^+$ and $\epsilon(t) = e^{-\beta(t)}$ in which minimum $f(\cdot)$ is sought. It is a straightforward job to verify that both BSA and LSA algorithms can be represented by the above form, from which their optimalities easily follow.

On the other hand, it turns out that the transition probabilities of the RSA algorithm does not conform to Eq. (3), and thus their analysis cannot be immediately applied to show its optimality. Nevertheless, we obtain the following result.

Theorem 1: RSA algorithm is annealing optimal.

The major part of the analysis is to generalize the transition probability form of Eq. (3) in order to represent multiple conditional transition probabilities of RSA algorithm for different message acquisition events, and to verify a suitably defined notion of *recurrence order* of each state, which conceptually captures how likely the system tends to stay on the state in a certain asymptotic sense, remains the same as that of BSA algorithm albeit the generalization. For brevity of the presentation, we provide the detailed steps for the proof in Appendix.

C. Efficiency in asymptotic variance rate

We now provide an insight into understanding the benefit of the proposed approaches by comparing different algorithms: BSA, LSA, and RSA algorithms. To quantitatively analyze and compare these algorithms, we first need to choose a specific metric that characterizes one algorithm being a good one.

One popular metric often considered in the literature is the mixing time. Conceptually, the mixing time of a Markov chain is the time until the Markov chain is close to its stationary state. In the standard Markov chain theory [14], the mixing time is precisely defined as

$$t_{mix}(\epsilon') = \min\{t \geq 1 : \max_{i \in \Omega} \|P_t(i, A) - \pi(A)\|_{\text{TV}} \leq \epsilon', \forall A \subseteq \Omega\},$$

which is the formalization of the idea: how large must t be until the time-t distribution is ϵ'-close to $\boldsymbol{\pi}$. Unfortunately, directly dealing with this quantity is a very difficult task, and most of existing analytic techniques rely on the spectral analysis based on the relation $t_{mix}(\epsilon') \leq \log(1/(\epsilon'\pi_{\min}))/(1 - \text{SLEM}(\mathbf{P}))$, where $\text{SLEM}(\mathbf{P}) = \max\{\eta_2, |\eta_{|\Omega|}|\}$ is the second largest eigenvalue modulus and $1 = \eta_1 \geq \ldots \geq \eta_{|\Omega|} \geq -1$ are the left eigenvalues of $\mathbf{P}$. Although the common wisdom in the literature is that the smaller SLEM is the smaller mixing time the chain $\mathbf{P}$ will have, its ordering relation on the upper bounds does not necessarily imply the chain with a smaller SLEM will actually mix faster in a rigorous sense.

Instead, we look at another performance metric that has been extensively used in the sampling theory. Sampling schemes are often used to estimate $\mathbb{E}_{\boldsymbol{\pi}}(h) \triangleq \sum_{i \in \Omega} h(i)\pi(i)$

for various functionals $h : \Omega \to \mathbb{R}$ by generating t samples $\{X(s)\}_{s=1}^{t}$ and constructing an estimator $\hat{\mu}_t(h) = \frac{1}{t}\sum_{s=1}^{t} h(X(s))$. In assessing the accuracy of this estimator, the asymptotic variance rate has been used as an important criterion in the literature. The asymptotic variance rate $\sigma(\mathbf{P}, h)$ of the estimate $\hat{\mu}_t(h)$ is defined in [14] as

$$\sigma(\mathbf{P}, h) = \lim_{t \to \infty} t \cdot \mathrm{Var}(\hat{\mu}_t(h)). \tag{4}$$

It has been known that the quantity $\sqrt{t}(\hat{\mu}_t(h) - \mathbb{E}_{\boldsymbol{\pi}}(h))$ converges in distribution to a Gaussian random variable with zero mean and variance $\sigma(\mathbf{P}, h)$.

We consider the ordering relationship among the three algorithms in term of the asymptotic variance rate. A useful technique related to this task is the so-called *Peskun ordering*, which is described next.

Definition 2 (Peskun ordering): [19] For two finite irreducible Markov chains on a finite state space Ω with $\mathbf{P} = \{P_{ij}\}_{i,j\in\Omega}$ and $\mathbf{P}' = \{P'_{ij}\}_{i,j\in\Omega}$ with the same stationary distribution $\boldsymbol{\pi}$, it is said that $\mathbf{P}'$ dominates $\mathbf{P}$ off the diagonal, written as $\mathbf{P} \preceq \mathbf{P}'$ if $P_{ij} \leq P'_{ij}$ for all $i, j \in \Omega$ ($i \neq j$).

Lemma 1: [19] If $\mathbf{P}$ and $\mathbf{P}'$ are reversible with respect to $\boldsymbol{\pi}$, and $\mathbf{P} \preceq \mathbf{P}'$, then $\sigma(\mathbf{P}, h) \geq \sigma(\mathbf{P}', h)$ for any h with $\mathrm{Var}_{\boldsymbol{\pi}}(h) \triangleq \sum_{i\in\Omega}(h(i)\pi(i) - \mathbb{E}_{\boldsymbol{\pi}}(h))^2 < \infty$.

Note from Proposition 1 that the stationary distributions of LSA and BSA algorithms are identical as $\boldsymbol{\pi}$ in Eq. (1). Also, the relation $\mathbf{P}^B(\mathbf{x}, \mathbf{x}') \geq \mathbf{P}^L(\mathbf{x}, \mathbf{x}')$ for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$) in Proposition 2 is exactly the definition of Peskun ordering. Therefore, we obtain the following consequence.

Proposition 3: $\sigma(\mathbf{P}^L, h) \geq \sigma(\mathbf{P}^B, h)$, for any h with $\mathrm{Var}_{\boldsymbol{\pi}}(h) < \infty$.

However, the stationary distribution of RSA algorithm, denoted by $\boldsymbol{\pi}^R$, is not necessarily equivalent to $\boldsymbol{\pi}$. For this reason, we cannot rely on the Peskun ordering relation between RSA and the others. Unfortunately, the efficiency analysis for comparing two Markov chains with different stationary distributions is notoriously difficult, and to the best of our knowledge there are no known technical tools applicable to our case. We leave the following statement as our conjecture.

Conjecture 1: $\sigma(\mathbf{P}^L, h) \geq \sigma(\mathbf{P}^R, h) \geq \sigma(\mathbf{P}^B, h)$, for any h with $\mathrm{Var}_{\boldsymbol{\pi}}(h) < \infty$ and $\mathrm{Var}_{\boldsymbol{\pi}^R}(h) < \infty$.

The rationale for the conjecture is that we observed from various simulations that the distributional bias of RSA algorithm is often very small, and hence we expect from Proposition 2 that a similar ordering relation will hold. This conjecture is empirically found to be true in diverse cases.

VI. NUMERICAL EVALUATION

In this section, we present the numerical experiments for the proposed algorithms. We first consider the network scenario of the four link case presented in Section V, where simulations are performed with using fixed but different temperature parameters in order to observe how different algorithms behave in a specific temperature regime. We assume that the weight parameters, $\{w_i\}$, are given and fixed as such described in the earlier section, and each node knows these parameters
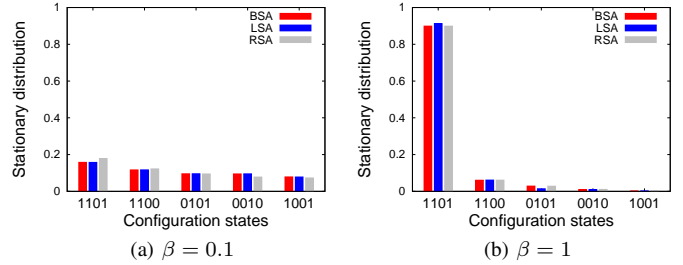

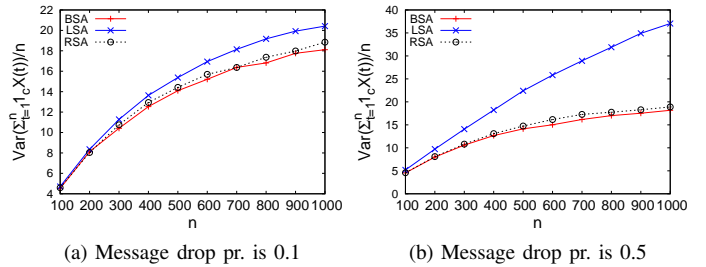Fig. 2. Comparison of different algorithms in their stationary distributions with different β.

Fig. 3. The variance rate of the link c's service process over different time scales ($\beta = 0.5$).

for all of its neighbor nodes. And, we chose $b^{ij}_{x_i, x'_i}$ is 0 for $x_i = 1, x'_i = 0$, and is $-w_j$ for $x_i = 0, x'_i = 1$, for all i and $j \in N_i$ for the bound parameters. Fig 2-A and 2-B plot the stationary distributions obtained from different algorithms for $\beta = 0.1$ and $\beta = 1$, respectively. Messages generated from neighbor nodes are set to be lost independently with probability 0.5 for LSA and RSA algorithms. Note that the results of BSA algorithm corresponds to those of LSA (or RSA) algorithm with no message loss events. As expected from the traditional analysis of SA and the form of Gibbs distribution, it can be seen from the figure that the distribution from BSA algorithm is scattered around different states for small β ($\beta = 0.1$), whereas it becomes concentrated on the optimal state, $(1, 1, 0, 1)$, for large β ($\beta = 1$). Similarly, the results of both LSA and RSA algorithms also match well with them, which reveals that the same annealing optimality will hold for RSA algorithm as well.

In Fig 3, we plot the variance rate defined in Eq. (4) with the choice of $h(X(t)) = \mathbf{1}_c(X(t)) \triangleq \mathbf{1}\{X(t) = (0, 0, 1, 0)\}$ in order to look at the variability of the cumulative service process of link c over different time scales. In this case, $\beta = 0.5$ is used. The figure 3-(a) shows the result with the message drop probability 0.1, in which the variability of link service process due to LSA algorithm increases quite a bit, whereas the result from RSA algorithm is almost same as that of BSA algorithm. When the message drop rate is high (as in Fig 3-(b)), the performance loss due to LSA algorithm is quite significant, whereas RSA algorithm can still be performed *as if there is no message drop* in this case. These results are consistent with our expectation described in Conjecture 1.

We here consider a queueing application scenario, where each node is associated with queue fed by an external packet arrival process. Let $Q_i(t)$ be the queue size of node i at time
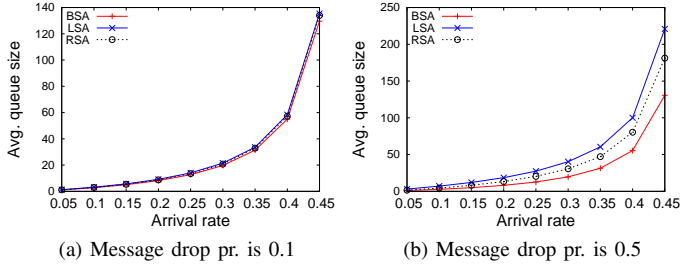
(a) Message drop pr. is 0.1   (b) Message drop pr. is 0.5

Fig. 4.   Average queue size of link c

Fig. 5.   Network topology. Arrows represent communication links, and red-dotted lines indicate the message collection structure.

SINR (dB)	Data rate (units per slot)
$(-\infty, 10]$	0
$(10, 20]$	1
$(20, 30]$	2
$(30, \infty)$	3

TABLE I
DATA RATES AS A FUNCTION OF SINR

(a) Message drop pr. is 0.1   (b) Message drop pr. is 0.3

Fig. 6.   Average queue size of the red-circled link in Fig 5.

t of which dynamics is determined by the typical queueing process: $Q_i(t) = [Q_i(t-1) + a_i(t) - c_i(\mathbf{x}(t))]^+$, $t \geq 1$, where the arrival process $a_i(t)$ is assumed to be a constant a_i over time. In order to adapt to the dynamic queue size, we adopt the popular technique of *dynamic fugacity* scheme used in the CSMA scheduling, in which the weight parameters are chosen such that $W_i(t) = \log(Q_i(t)+1)$. The parameter β is simply set to be 1, since setting large weight parameters (or equivalently large queue size) will act as setting large β in the max-weight problem, and the appropriate weights will be automatically found from the queue sizes. Now the weight parameters are dynamic over time, and therefore they should also be informed to the corresponding nodes as a form of message, which is also subject to the delivery failure events. Since the queue dynamics has a limited evolution over time: $Q_i(t) + a_i k - k \leq Q_i(t+k) \leq Q(t) + a_i k$, $k \geq 0$, each node i can obtain an upper bounded estimate for the weights $w_j(t+k)$ for $j \in N_i$, based on the most recently observed value of $w_j(t)$. We have implemented these schemes in the simulation, and plotted the average queue size of link c in Fig 4. The results show that the improvement of average queue size by RSA algorithm can be quite significant especially when the message drop rate is high.

We also perform simulations with a realistic SINR model in a larger network with 10 pairs of communication links randomly deployed in a $500m \times 500m$ geographical region as shwon in Fig 5. The parameter settings are as following. The transmitter node of each link i has three transmission power modes, $x_i \in \{0, 5, 10\}$mW, and the transmitted signal experiences pass loss by $d_{ij}^{-\gamma}$, where $\gamma = 4$ and d_{ij} is the distance between the transmitter i and the receiver j of the signal, and the thermal noise $n_0 = 10^{-12}$mW is used for all links. Each link obtains different amount of capacity depending on its experienced SINR level as described in Table I, and the bound parameters are chosen with $c_i^{\max} = 3, \forall i$ and in the way we described in Section V-A. We have selected the set of neighbors from which the messages are to be collected during the messaging phase such that a link j belongs to link i's neighbor set N_i if the receiver node of link j is located within the range of 250m of the transmitter node of link i (denoted as red-dotted lines in Fig 5). The packets are injected uniformly to every queue with varying rates from 0.05 to which any queue gets saturated. We observe that the link with

---

The log function is used to effectively emulate the time scale separation assumption in a large queue regime which is a standard technique [22], [8].
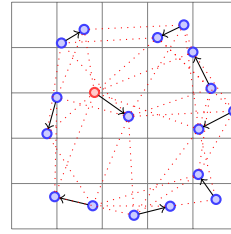
red-circled node tends to have the largest queue among all links, and hence we look at its queue size in the following.

The average queue size of the link with the message drop probabilities is plotted in Fig 6. We observe that the queueing performance of LSA algorithm is very sensitive to the message drop events, and it can be shown in Fig 6-(a) that its queue size quickly grows even with very small drop probability, where the performance of RSA algorithm is fairly close to the one without any message drop event. Fig 6-(b) shows that as the drop probability increases, the performance of RSA algorithm also gets worse, however, its improvement is substantial in comparison to LSA algorithm.

We also look at the impact of using loose bounds, rather than using the known-tight bound. For this, we intentionally used large values of $c_i^{\max} \in [3, \ldots, 10]$ for the bound parameters, and performed the simulations. Fig 7 plots the ratio of the measured queue size between RSA and LSA algorithm with using different bound parameters under the different message drop regimes (from 0.05 to 0.4). It shows that in both cases of light and heavy packets arrival intensity, the improvement is remarkable even with using twice larger bound than the tight bound when e.g., drop rate is 0.2 (the ratio in this case is 0.85 and 0.78 for the arrival rate 0.2 and 0.4 respectively), and in general the improvement by RSA algorithm can be seen substantial unless the bound is too loose.

VII. DISCUSSION

In this section, we discuss the relationship of our proposed approach with existing solutions from a practical viewpoint.

We remark that our implementation structure is comparable to those in [2] and [21]. While the goal of achieving the desired stationary distribution is the same, their implementation is a little different from ours. Their schemes are close to a
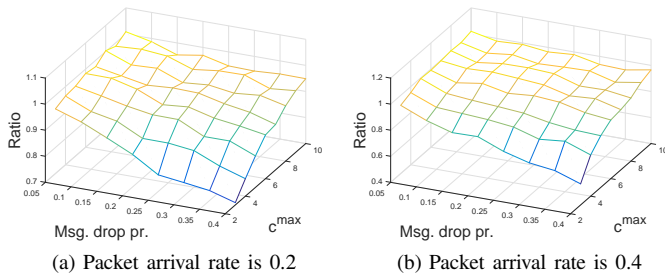
(a) Packet arrival rate is 0.2   (b) Packet arrival rate is 0.4

Fig. 7. The ratio of the averge queue size between RSA and LSA algorithms (RSA/LSA) with using different bounds in the various message drop regimes.

proactive approach in the following sense. Each node evaluates the performance objective of others based on its locally stored variable and prior knowledge on their functionals. Whenever a node changes its state, it proactively broadcasts (within a suitably defined local range) to other nodes to convey the new state, and those local variables can be updated upon receiving the message. On the other hand, our approach can be seen as a *reactive* approach in that whenever a node wants to update its state, it sends out a request signal to other nodes, and then those nodes that received the signal reacts to the request by sending messages containing its functional difference as described in our algorithm.

We notice that the proactive approach has a few drawbacks. First, even in the one-hop interference model - a capacity model that only considers interference from nodes within a directly communicable range - the earlier mentioned broadcasting task has to be performed over a two-hop range, except for some special instances [2]. Also, the condition that each node has a prior knowledge on the objective function of others may not be a realistic assumption in practice. In the SINR model, this condition requires that all pairs of links have knowledge of the channel gain terms between the transmitter of their own link and the receiver of their neighboring links, which is difficult to know a priori.

Regarding the message overhead, we take no position on one approach being better than the other, as it will greatly differ depending on the type of network topology and how to realize those message exchange mechanisms with particular communication systems. Typically, broadcasting is easier than receiving different information from different nodes, however, it has been well known that ensuring the correct reception of broadcasting is a difficult job, which may incur additional overhead. Needless to say, doing that with two-hop broadcasting is even more difficult. To avoid this difficulty, [2] and [21] suggest using *out-dated* values, i.e., the most recently known values about the corresponding variables. However, the mismatch between the local variables and the actual ones will incur bias to the resulting stationary distribution, and thus whether it can achieve the same optimality is not clear. This fact has been neglected in those works.

In contrast, in our approach, in the one-hop interference model, we only require one-hop information, where its complexity can be robustly handled by the new approach we introduced in this paper. Furthermore, our algorithm does not need to know the precise form of the objective functions. Hence, our implementation approach is advantageous and should be applicable to communication systems that go even beyond the SINR model.

VIII. Conclusion

In this paper, we investigate important practical considerations and performance issues that arise when the traditional SA is implemented in wireless networks with imperfect communications. We recognize that various practical factors including the inherently noisy nature of wireless communications and the increasing message complexity in modern communication technologies can be critical sources that prevent the efficient realization of SA in general wireless networks. Our simulation results show that a straightforward solution to bypass this problem is not practical due to its slow operation speed. To tackle this problem, we propose a novel approach that allows the algorithm to operate with only partial observations on the system performance objective, which helps improve its computational speeds. We rigorously show that the new algorithm exhibits the same convergence in probability to the optimal states under the standard annealing technique.

References

[1] F. Bacelli and P. Brémaud. *Elements of Queueing Theory: Palm Martingale Calculus and Stochastic Recurrences*. Springer-Verlag, 2003.

[2] S. Borst, M. Markakis, and I. Saniee. Distributed power allocation and user assignment in ofdma cellular networks. In *IEEE Allerton Conference*, 2011.

[3] D. P. Connors and P. Kumar. Balance of recurrece order in time-inhomogenous markov chains with application to simulated annealing. *Probability in the Engineering and Informational Sciences*, 2(02):157–184, 1988.

[4] D. P. Connors and P. Kumar. Simulated annealing type markov chains and their order balance equations. *SIAM Journal on Control and Optimization*, 27(6):1440–1461, 1989.

[5] M. Desai, S. Kumar, and P. Kumar. Quasi-statically cooled markov chains. *Probability in the Engineering and Informational Sciences*, 8(1):1–19, 1994.

[6] A. Eryilmaz, E. Modiano, and A. Ozdaglar. Randomized algorithms for throughput-optimality and fairness in wireless networks. In *IEEE CDC*, 2006.

[7] A. Ganesh, N. O'Connell, and D. Wischik. *Big Queues*. Springer, 2004.

[8] J. Ghaderi and R. Srikant. On the design of efficient csma algorithms for wireless networks. In *IEEE CDC*, 2010.

[9] M. Gong, L. Ma, Q. Zhang, and L. Jiao. Community detection in networks by using multiobjective evolutionary algorithm with decomposition. *Physica A: Statistical Mechanics and its Applications*, 391(15):4050–4060, 2012.

[10] B. Hajek. Cooling schedules for optimal annealing. *Mathematics of operations research*, 13(2):311–329, 1988.

[11] L. Jiang and J. Walrand. A distributed csma algorithm for throughput and utility maximization in wireless networks. *IEEE/ACM Trans. on Networking*, 18(3):960–972, 2010.

[12] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, et al. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[13] H. W. Lee, E. Modiano, and L. B. Le. Distributed throughput maximization in wireless networks via random power allocation. *IEEE Transactions on Mobile Computing*, 11(4):577–590, April 2012.

[14] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2009.

[15] E. Modiano, D. Shah, and G. Zussman. Maximizing throughput in wireless networks via gossiping. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '06/Performance '06. ACM, 2006.

[16] M. J. Neely, E. Modiano, and C. E. Rohrs. Power allocation and routing in multibeam satellites with time-varying channels. *IEEE/ACM Transactions on Networking (TON)*, 11(1):138–152, 2003.

[17] J. Ni, B. Tan, and R. Srikant. Q-csma: Queue-length based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks. In *IEEE INFOCOM*, 2010.

[18] F. Ogbu and D. K. Smith. The application of the simulated annealing algorithm to the solution of the n/m/cmax flowshop problem. *Computers & Operations Research*, 17(3):243–253, 1990.

[19] P. H. Peskun. Optimum monte-carlo sampling using markov chains. *Biometrika*, 60(3):607–612, 1973.

[20] D. Qian, D. Zheng, J. Zhang, N. B. Shroff, and C. Joo. Distributed csma algorithms for link scheduling in multihop mimo networks under sinr model. *IEEE/ACM Transactions on Networking*, 21(3):746–759, 2013.

[21] L. P. Qian, Y. J. Zhang, and M. Chiang. Globally optimal distributed power control for nonconcave utility maximization. In *IEEE GLOBE-COM*, 2010.

[22] S. Rajagopalan, D. Shah, and J. Shin. Network adiabatic theorem: An efficient randomized protocol for contention resolution. In *ACM SIGMETRICS/Performance*, 2009.

[23] P. S. Swamy, R. K. Ganti, and K. Jagannathan. Adaptive csma under the sinr model: Efficient approximation algorithms for throughput and utility maximization. *IEEE/ACM Transactions on Networking*, 2017.

[24] P. Tian, J. Ma, and D.-M. Zhang. Application of the simulated annealing algorithm to the combinatorial optimisation problem with permutation property: An investigation of generation mechanism. *European Journal of Operational Research*, 118(1):81–94, 1999.

[25] S. Vasudevan, M. Adler, D. Goeckel, and D. Towsley. Efficient algorithms for neighbor discovery in wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 21(1):69–83, 2013.

APPENDIX

In this appendix, we provide proofs for Proposition 1, 2, and Theorem 1. To that end, we first write down the transition probabilities of BSA, LSA and RSA algorithms, respectively denoted by $\mathbf{P}^B$, $\mathbf{P}^L$, and $\mathbf{P}^R$, to be referred in the proofs. For notational simplicity, we denote for two configurations $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ which differ only at one node $i \in \mathcal{N}$, i.e., $x_i \neq x'_i$ and $x_j = x'_j$ for all $j \in \mathcal{N} \setminus \{i\}$, that $\hat{i}(\mathbf{x}, \mathbf{x}')$ (or simply $\hat{i}$ when its definition is clear) indicates the node with the different state.

The transition probability of BSA algorithm, $P^B(\mathbf{x}, \mathbf{x}')$, for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$), is

$$P^B(\mathbf{x}, \mathbf{x}') = c(\mathbf{x}, \mathbf{x}')e^{-\beta[-\Delta(\mathbf{x}, \mathbf{x}')]^+}, \qquad (5)$$

where $\Delta(\mathbf{x}, \mathbf{x}') = \sum_{j \in N_{\hat{i}} \cup \{\hat{i}\}} f_j(\mathbf{x}') - f_j(\mathbf{x})$ and

$$c(\mathbf{x}, \mathbf{x}') = \begin{cases} \frac{1}{n(|\mathcal{M}|-1)}, & \text{if for some } i \in \mathcal{N}, \ x_i = x'_i \text{ and,} \\ & x_j \neq x'_j, \forall j \in N_i \setminus \{i\} \\ 0, & \text{if otherwise.} \end{cases}$$

The transition probability of LSA algorithm, $P^L(\mathbf{x}, \mathbf{x}')$, for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$), is

$$P^L(\mathbf{x}, \mathbf{x}') = c(\mathbf{x}, \mathbf{x}')q_{\hat{i}, N_{\hat{i}}}e^{-\beta[-\Delta(\mathbf{x}, \mathbf{x}')]^+}. \qquad (6)$$

The transition probability of RSA algorithm $P^R(\mathbf{x}, \mathbf{x}')$, for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$), is

$$P^R(\mathbf{x}, \mathbf{x}') = c(\mathbf{x}, \mathbf{x}') \sum_{S \subseteq N_{\hat{i}}} q_{\hat{i}, S}e^{-\beta[-\Delta_{[S]}(\mathbf{x}, \mathbf{x}')]^+} \qquad (7)$$

where

$$\Delta_{[S]}(\mathbf{x}, \mathbf{x}') = \sum_{j \in S \cup \{\hat{i}\}} f_j(\mathbf{x}') - f_j(\mathbf{x}) + \sum_{j \in N_{\hat{i}} \setminus \{S\}} b^{\hat{i}j}_{x_{\hat{i}}, x'_{\hat{i}}}.$$

For all three algorithms, their self transition probabilities are obtained by $P^B(\mathbf{x}, \mathbf{x}) = 1 - \sum_{\mathbf{x}' \neq \mathbf{x}} P^B(\mathbf{x}, \mathbf{x}')$ (similarly for P^L and P^R), for all $\mathbf{x} \in \mathcal{X}$.

Proof of Proposition 1.

Proof: Let $\boldsymbol{\pi}^L$ be the stationary distribution of LSA algorithm. Then, for any $\mathbf{x}, \mathbf{x}' \in \Omega$ ($\mathbf{x} \neq \mathbf{x}'$), it holds

$$\pi^L(\mathbf{x})P^L(\mathbf{x}, \mathbf{x}') = \pi^L(\mathbf{x}')P^L(\mathbf{x}', \mathbf{x})$$
$$\Leftrightarrow \quad \pi^L(\mathbf{x})P^B(\mathbf{x}, \mathbf{x}') = \pi^L(\mathbf{x}')P^B(\mathbf{x}', \mathbf{x}).$$

Since $\boldsymbol{\pi}$ is the unique solution of $\boldsymbol{\pi}^L$ in the above set of equations along with the probability constraint, $\sum_{\mathbf{x}} \pi^L(\mathbf{x}) = 1$, it follows $\boldsymbol{\pi} = \boldsymbol{\pi}^L$. ∎

Proof of Proposition 2.

Proof: Note that for any $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ in which only one node state is different, and for any $S \subseteq N_{\hat{i}}$, the following holds

$$\begin{aligned} \Delta(\mathbf{x}, \mathbf{x}') &= \Delta_{[N_{\hat{i}}]}(\mathbf{x}, \mathbf{x}') \\ &= \sum_{j \in N_i \cup \{\hat{i}\}} f_j(x'_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) - f_j(x_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) \\ &\geq \sum_{j \in S \cup \{\hat{i}\}} f_j(x'_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) - f_j(x_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) \\ &\quad + \sum_{j \in N_i \setminus \{S\}} \left(\min_{\mathbf{x}_{[-i]}} f_j(x'_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) - \max_{\mathbf{x}_{[-\hat{i}]}} f_j(x_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) \right) \\ &\geq \sum_{j \in S \cup \{\hat{i}\}} f_j(x'_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) - f_j(x_{\hat{i}}, \mathbf{x}_{[-\hat{i}]}) + \sum_{j \in N_i \setminus S} b^{\hat{i}j}_{x_{\hat{i}}, x'_{\hat{i}}} \\ &= \Delta_{[S]}(\mathbf{x}, \mathbf{x}'), \qquad (8) \end{aligned}$$

and therefore we have

$$e^{-\beta[-\Delta_{[S]}(\mathbf{x}, \mathbf{x}')]^+} \leq e^{-\beta[-\Delta(\mathbf{x}, \mathbf{x}')]^+}. \qquad (9)$$

Since $q_{i, N_i} \leq \sum_{S \subseteq N_i} q_{i, S} = 1$ holds for all $i \in \mathcal{N}$, the statement follows from the transition probabilitiy of each algorithm in Eq. (5-7) and the inequality in Eq. (9). ∎

Proof of Theorem 1.

Proof: Our proof for the theorem is based on the technique introduced in [4], in which the annealing optimality of the original SA is proven. We first briefly overview the major steps therein, and apply them to show the optimality of RSA algorithm.

The authors in [4] consider a class of Markov chains whose transition probabilities can be written as a form of Eq. (3), in which the conventional simulated annealing algorithm can be represented by setting $V_{ij} = [f(j) - f(i)]^+$ and $\epsilon(t) = e^{-\beta(t)}$ for achieving minimum $f(\cdot)$. For this class of Markov chains, they define the *recurrence order* for each state and transition of the Markov chain as follows.

Definition 3: The order of recurrence of a state $i \in \Omega$, denoted α_i, is

$$\alpha_i := \begin{cases} -\infty, & \text{if } \sum_{t=1}^{\infty} \mu_i(t) < \infty, \\ p^-, & \text{if } p = \sup\{c \geq 0 : \sum_{t=1}^{\infty} \epsilon(t)^c \mu_i(t) = \infty\} \\ & \quad \text{and } \sum_{t=1}^{\infty} \epsilon(t)^p \mu_i(t) < \infty, \\ p & \text{if } p = \max\{c \geq 0 : \sum_{t=1}^{\infty} \epsilon(t)^c \mu_i(t) = \infty\} \end{cases}$$

where $\mu_i(t) = \mathbb{P}\{X(t) = i\}$ and p is regarded as strictly larger than p^-, i.e., $p > p^- > p - \delta_0$ for some $\delta_0 > 0$. Similarly, the *order of recurrence of the transition from i to j* is defined by,

Definition 4: The order of recurrence of the transition from i to j, denoted α_{ij}, is

$$\alpha_{ij} := \begin{cases} -\infty, & \text{if } \sum_{t=1}^{\infty} \mu_{ij}(t) < \infty, \\ p^-, & \text{if } p = \sup\{c \geq 0 : \sum_{t=1}^{\infty} \epsilon(t)^c \mu_{ij}(t) = \infty\} \\ & \quad \text{and } \sum_{t=1}^{\infty} \epsilon(t)^p \mu_{ij}(t) < \infty, \\ p & \text{if } p = \max\{c \geq 0 : \sum_{t=1}^{\infty} \epsilon(t)^c \mu_{ij}(t) = \infty\} \end{cases}$$

where $\mu_{ij}(t) = \mathbb{P}\{X(t) = i, X(t+1) = j\}$. They also defined ρ, the *order of cooling* of $\{\epsilon(t)\}$, as follows.

Definition 5: The order of the cooling schedule $\{\epsilon(t)\}$, denoted ρ, is defined as

$$\rho := \begin{cases} -\infty, & \text{if } \sum_{t=1}^{\infty} \epsilon(t) < \infty, \\ p^-, & \text{if } p = \sup\{c \geq 0 : \sum_{t=1}^{\infty} \epsilon(t)^c = \infty\} \\ & \quad \text{and } \sum_{t=1}^{\infty} \epsilon(t)^p < \infty, \\ p & \text{if } p = \max\{c \geq 0 : \sum_{t=1}^{\infty} \epsilon(t)^c = \infty\} \end{cases}$$

Having defined the above terms, we summarize the main results established in [4] as well as in [3], [5], which are valid under the following mild assumptions.

Assumptions

1) d is sufficiently large. In particular, $d \geq 2\sum_{(i,j)|V_{ij}<\infty} V_{ij}$.
2) $\exists j \in N_i \Leftrightarrow \exists i \in N_j \ \forall i, j \in \Omega$.

Lemma 2: [4], [3], [5] Under the above assumptions, the followings hold.

1) The relation between α_i and α_{ij} is

$$\alpha_{ij} = \alpha_i - V_{ij} \quad \text{for all } i, j \in \Omega \ (i \neq j), \quad (10)$$

2) There is a balance of recurrence orders across every edge in the graph of the Markov chain such that

$$\max_{i \in A, j \in A^c} \alpha_{ij} = \max_{i \in A, j \in A^c} \alpha_{ji} \quad \text{for all } A \subseteq \Omega. \quad (11)$$

3) $\{\alpha_i\}$ is the unique solution $\{\lambda_i\}$ of

$$\max_{i \in A, j \in A^c} \lambda_i - V_{ij} = \max_{i \in A, j \in A^c} \lambda_j - V_{ji} \quad \text{for all } A \subseteq \Omega,$$
$$\max_{i \in \Omega} \lambda_i = \rho.$$

4) Recall that d is the rate of the cooling schedule $\epsilon(t) = t^{-1/d}$, $t \geq 1$. It can be shown that

$$d = \max_{i \in \Omega} \alpha_i = \rho, \text{ and } \alpha_i = \rho \text{ iff } i \in \Omega^*,$$

where $\Omega^* = \{i \in \Omega : f(i) = \min_{j \in \Omega} f(j)\}$.

5) Let $\Omega^\dagger$ be the set of states of the largest recurrence order, i.e., $\Omega^\dagger := \{i \in \Omega : \alpha_i = \max_{j \in \Omega} \alpha_j\}$. Then,

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{P}\{X(t) \in \Omega^\dagger\} = 1. \quad (12)$$

Note that the transition probability of RSA algorithm in Eq. (7) cannot be represent by Eq. (3). To pursue the above approach to verifying the optimality of RSA algorithm, it is necessary to consider a more general class of Markov chains which has the transition probabilities of the form,

$$p_{ij}(t) = c_{ij} \sum_{m \in M^{ij}} a_{ij}^m \epsilon(t)^{V_{ij}^m}, \quad (13)$$

where M^{ij} is some finite set associated with the transition from i to j, $i, j \in \Omega$, and $a_{ij}^m \in (0, 1]$ is a probability of an element $m \in M^{ij}$ defined over its sample space M^{ij}, and $V_{ij}^m \geq 0$ for all $m \in M^{ij}$. Now the transition probability of RSA algorithm can be represented by Eq. (13) by replacing the corresponding terms: $\Omega \Leftrightarrow \mathcal{X}$, $i, j \in \Omega \Leftrightarrow \mathbf{x}, \mathbf{x}' \in \mathcal{X}$, $c_{ij} \Leftrightarrow c(\mathbf{x}, \mathbf{x}')$, $M^{ij} \Leftrightarrow S(\mathbf{x}, \mathbf{x}') := \{S \in \mathcal{P}(N_{i'}) : q_{\hat{i},S} > 0\}$, $m \in M^{ij} \Leftrightarrow S \in S(\mathbf{x}, \mathbf{x}')$, $a_{ij}^m \Leftrightarrow q_{\hat{i},S}$, $V_{ij}^m \Leftrightarrow V^S(\mathbf{x}, \mathbf{x}') := \Delta_{[S]}(\mathbf{x}, \mathbf{x}')$, where $\mathcal{P}(A)$ is the powerset of a set A. For this form, we obtain the generalized correspondence of eq. (10) as stated in the following lemma.

Lemma 3: $\alpha_{ij} = \alpha_i - \max_{m \in M^{ij}} V_{ij}^m, \quad \forall i, j \in \Omega$.

Proof: By the Chapman-Kolmogorov equation, we obtain

$$\mu_{ij}(t) = c_{ij} \sum_{m \in M^{ij}} a_{ij}^m \epsilon(t)^{V_{ij}^m} \mu_i(t)$$

and observe that

$$\sum_{t=1}^{\infty} \sum_{m \in M^{ij}} a_{ij}^m \epsilon(t)^{c+V_{ij}^m} \mu_i(t) = \infty$$

holds if and only if there exists some $m \in M^{ij}$ such that

$$\sum_{t=1}^{\infty} \epsilon(t)^{c+V_{ij}^m} \mu_i(t) = \infty.$$

Therefore, we have

$$\sup\{c \geq 0 : \sum_{t=1}^{\infty} \sum_{m \in M^{ij}} a_{ij}^m \epsilon(t)^{c+V_{ij}^m} \mu_i(t) = \infty\}$$
$$= \sup\{c \geq 0 : \sum_{t=1}^{\infty} \epsilon(t)^{c+\max_{m \in M^{ij}} V_{ij}^m} \mu_i(t) = \infty\}$$

from which the result follows by the definitions of α_i and α_{ij}. ∎

Let $\alpha(\mathbf{x})$ and $\alpha(\mathbf{x}, \mathbf{x}')$ be the recurrence order of state $\mathbf{x}$ and that of state transition from $\mathbf{x}$ to $\mathbf{x}'$, for $\mathbf{x}, \mathbf{x}' \in \Omega \ (\mathbf{x} \neq \mathbf{x}')$ due to the Markov chain induced by BSA Algorithm, and let $\hat{\alpha}(\mathbf{x})$ and $\hat{\alpha}(\mathbf{x}, \mathbf{x}')$ be those of the chain from RSA algorithm, respectively. In the following lemmas, we verify that all the recurrence orders for any state and transition are equivalent between the two algorithms.

In [4], [3], the convergence analysis is only conducted for showing $\limsup_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{P}\{X(t) \in \Omega^\dagger\} = 1$, for a general class of cooling schedules, however it can be shown that the limit holds for the particular cooling schedule $\epsilon(t) = t^{-1/d}$ [5].

Lemma 4: $\max_{\mathbf{x} \in \mathcal{X}} \hat{\alpha}(\mathbf{x}) = \rho = d$.

Proof: Note that $\sum_{t=1}^{\infty} \epsilon(t)^c < \infty$ for $c > d$ and $\sum_{t=1}^{\infty} \epsilon(t)^c = \infty$ for $c \leq d$, and thus by the definition of ρ, $d = \rho$. On the other hand,

$$\sum_{t=1}^{\infty} \epsilon(t)^d = \sum_{\mathbf{x} \in \mathcal{X}} \left(\sum_{t=1}^{\infty} \epsilon(t)^d \mu_{\mathbf{x}}(t) \right) = \infty,$$

where $\mu_{\mathbf{x}}(t) = \mathbb{P}\{X(t) = \mathbf{x}\}$, implying $\sum_{t=1}^{\infty} \epsilon(t)^d \mu_{\mathbf{x}}(t) = \infty$ for some $\mathbf{x} \in \mathcal{X}$. Therefore, $\max_{\mathbf{x} \in \mathcal{X}} \alpha(\mathbf{x}) = d$, and the same analysis also applies to $\hat{\alpha}(\mathbf{x})$. ∎

Lemma 5: $\alpha(\mathbf{x}) = \hat{\alpha}(\mathbf{x})$, and $\alpha(\mathbf{x}, \mathbf{x}') = \hat{\alpha}(\mathbf{x}, \mathbf{x}')$, for all $\mathbf{x}, \mathbf{x}' \in \Omega$,

Proof: Note that the order balance equation of Eq. (11) for the BSA algorithm can be written by

$$\max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \alpha(\mathbf{x}, \mathbf{x}') = \max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \alpha(\mathbf{x}', \mathbf{x}), \quad \forall A \subseteq \mathcal{X},$$

and $\{\alpha(\mathbf{x})\}$ is the unique solution of $\{\lambda(\mathbf{x})\}$ of the equations

$$\max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \lambda(\mathbf{x}) - V^{N_{\hat{i}}}(\mathbf{x}, \mathbf{x}') = \max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \lambda(\mathbf{x}') - V^{N_{\hat{i}}}(\mathbf{x}', \mathbf{x}),$$

$$(14)$$

for all $A \subseteq \mathcal{X}$ along with $\max_{\mathbf{x} \in \mathcal{X}} \lambda(\mathbf{x}) = \rho$. On the other hand, a similar characterization of $\{\hat{\alpha}(\mathbf{x})\}$ of RSA algorithm using the order balance equation,

$$\max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \hat{\alpha}(\mathbf{x}, \mathbf{x}') = \max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \hat{\alpha}(\mathbf{x}', \mathbf{x}), \quad \forall A \subseteq \mathcal{X},$$

can be written (from Lemma 3) as the solution $\{\hat{\lambda}(\mathbf{x})\}$ of

$$\max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \left(\hat{\lambda}(\mathbf{x}) - \max_{S \subseteq S(\mathbf{x}, \mathbf{x}')} V^S(\mathbf{x}, \mathbf{x}') \right)$$

$$= \max_{\mathbf{x} \in A, \mathbf{x}' \in A^c} \left(\hat{\lambda}(\mathbf{x}') - \max_{S \subseteq S(\mathbf{x}', \mathbf{x})} V^S(\mathbf{x}', \mathbf{x}) \right), \quad (15)$$

for all $A \subseteq \mathcal{X}$ with $\max_{\mathbf{x} \in \mathcal{X}} \hat{\lambda}(\mathbf{x}) = \rho$ (due to Lemma 4). Observe that for any $S \subseteq S(\mathbf{x}, \mathbf{x}')$, $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$), it holds

$$V^{N_{\hat{i}}}(\mathbf{x}, \mathbf{x}') = \Delta_{[N_{\hat{i}}]}(\mathbf{x}, \mathbf{x}') \geq \Delta_{[S]}(\mathbf{x}, \mathbf{x}') = V^S(\mathbf{x}, \mathbf{x}'),$$

where the inequality is from Eq. (8). Since $q_{i, N_i} > 0$ for all $i \in \mathcal{N}$, we have

$$V^{N_{\hat{i}}}(\mathbf{x}, \mathbf{x}') = \max_{S \in S(\mathbf{x}, \mathbf{x}')} V^S(\mathbf{x}, \mathbf{x}'),$$

and hence,

$$\hat{\alpha}(\mathbf{x}, \mathbf{x}') = \hat{\alpha}(\mathbf{x}) - \max_{S \in S(\mathbf{x}, \mathbf{x}')} V^S(\mathbf{x}, \mathbf{x}')$$

$$= \hat{\alpha}(\mathbf{x}) - V^{N_{\hat{i}}}(\mathbf{x}, \mathbf{x}'), \quad (16)$$

for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ ($\mathbf{x} \neq \mathbf{x}'$), which presents the same forms in Eq. (14). Therefore, Lemma 2-3) implies that the (unique) solution for Eq. (15) is identical to that of Eq. (14), i.e., $\alpha(\mathbf{x}) = \hat{\alpha}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$. Also, $\alpha(\mathbf{x}, \mathbf{x}') = \hat{\alpha}(\mathbf{x}, \mathbf{x}')$ holds for all $\mathbf{x}, \mathbf{x}' \in \Omega$ ($\mathbf{x} \neq \mathbf{x}'$) from eq. (16). ∎

The result follows from the above lemma and Lemma 2-4) and 2-5). ∎