Bounded Length Least-Cost Path Estimation in Wireless Sensor Networks Using Petri Nets

Lingxi Li and Dongsoo Stephen Kim
Department of Electrical and Computer Engineering, Indiana University-Purdue University Indianapolis, Indiana, USA
Email: {LL7, dskim}@iupui.edu

Abstract— **This paper proposes an iterative algorithm for estimating the least-cost paths in a wireless sensor network (WSN) that is modeled by a Petri net. We assume that each link between two sensor nodes (each transition in the net) is associated with a positive cost to capture its likelihood (e.g., in terms of the transmission time, distance between nodes, or the link capacity). Given the structure of the Petri net and its initial/final state, we aim at finding the paths (with length less than or equal to a given bound L) that could lead us from the initial state to the final state while having the least total cost. We develop an iterative algorithm that obtains the least-cost paths with complexity that is polynomial in the length L.**

Index Terms— **Wireless sensor networks, Petri nets, least-cost, path estimation**

I. Introduction

In recent years, major advances in wireless communication and digital electronics have led to the development of sensors that can broadcast data over short communication ranges. A wireless sensor network (WSN) is composed of a number of sensor nodes that have sensing, processing, and communication capabilities that enable all sensor nodes to collaborate in order to achieve a particular task [1]. In a WSN, sensor nodes are densely deployed, have limited computational and power capacities, and use basic communication protocols to exchange information with their neighbors. In practice, WSNs have been widely employed in a variety of applications including transportation systems, military systems, and health-care systems [2]– [6].

As the size and complexity of WSNs increase, significant attention is paid to problems of modeling and analysis of WSNs. Since information transmission between sensor nodes can be treated as the occurrences of discrete events after some levels of abstraction, several discrete event dynamic system (DEDS) approaches have been proposed including finite state machines [7]– [9], Markov chains [10], [11], and Petri nets [12]– [17]. In literature, several aspects of problems related to WSNs (e.g., sensor data processing, energy consumption, and intelligent monitoring) have been studied using Petri net models due to their powerful modeling and mathematical analysis framework. In particular, the authors of [12] proposed a fuzzy Petri net model to process sensor data and handle flexible and continuous queries. In [13], the authors developed an attack-resistant and efficient localization scheme in WSNs based on Petri nets. The power consumption of processors in WSNs was studied using Petri nets in [14] and the energy consumption of wireless sensor nodes based on Petri net models was analyzed in [15]. A decentralized Petri net based wireless sensor node architecture (PN-WSNA) was developed to construct a flexible and reconfigurable WSN for intelligent monitoring systems in [16] and [17].

Other than the works mentioned above, in this paper we study a different problem in WSNs based on Petri net models, namely, the least-cost path estimation problem. In literature, path estimation and path planning in WSNs have been investigated using a variety of approaches. In [18], the authors considered the dynamic shortest path routing problem in mobile ad hoc networks. An genetic algorithm was developed to tackle this problem and was demonstrated its effectiveness through simulations. The authors of [19] studied the path planning problem in WSNs to deal with transmission delay using queueing networks. They proposed an approximate iterative algorithm to calculate node packet arrival rate and designed the pre-selection algorithm based on a path search tree. The optimal paths can then be obtained based on the analysis of the path delay. In [20], the authors proposed an shortest path routing protocol for mobile WSNs. This protocol was based on neighbor discovery and shortest path construction. The authors of [21] developed an optimal multi-path planning algorithm for intrusion detection in wireless sensor and actor networks. They associated each link with a cost that is related to packet transmission and employed graphical algorithms to determine the optimal paths. In [22], the authors considered the cluster routing problem in WSNs and proposed a fuzzy reasoning algorithm based on fuzzy Petri nets to improve the routing reliability and enhance the energy efficiency.

In this paper we study the least-cost path estimation problem in WSNs based on Petri net models. In particular, we consider a setting where we have knowledge of the structure of a Petri net and its initial state, and we are

This paper is based on "Least-cost path estimation in wireless ad hoc sensor networks using Petri nets," by L. Li and D. S. Kim, which appeared in the Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication (ICUIMC), Seoul, Korea, February 2011. © 2011 ACM.
Manuscript received May 28, 2011; revised July 10, 2011; accepted August 31, 2011.
This work was supported in part by the Indiana University-Purdue University Indianapolis (IUPUI) Research Support Funds Grant. Corresponding author: Lingxi Li, 723 W. Michigan St., SL 160, Indianapolis, IN 46202, USA. Email: LL7@iupui.edu.

doi:10.4304/jnw.7.1.81-87

given a final state (i.e., a target state to be reached) together with a positive integer number L which serves as the maximum allowable length for sequences of transitions whose firings can lead us from the initial state to the final state. We assume that each link in the WSN (each transition in the net) is associated with a positive cost that captures its likelihood (e.g., in terms of the transmission time, distance between nodes, or the link capacity). Given the above constraints (i.e., a Petri net structure, the initial/final state, and a length L), we aim at finding the paths in the WSN (transition firing sequences in the Petri net) with length less than or equal to L which: (i) are consistent with the structure of the Petri net, (ii) lead us from the initial state to the final state, and (iii) have the least total cost (the total cost of a transition firing sequence is taken to be the sum of the costs of all transitions in the sequence). We develop an iterative algorithm that obtains the least-cost transition firing sequences with complexity that is polynomial in L.

This work is motivated by the problem of least-cost path estimation in WSNs that are modeled by Petri nets. For example, in a WSN, information transmission involves the determination of detailed sequences of paths from an initial state to a final state. In our setup, the given final state captures the completion of the information transmission, whereas the structure of the given Petri net represents the interactions among different sensor nodes (as imposed by the given WSN). We also associate each transition in the given net with a positive cost which could represent its likelihood of occurrence. Note that in this paper, we focus on finding the least-cost transition firing sequences of length upper bounded by the given value L, which follows from the reasonable assumption of completing the information transmission in a WSN within a bounded number of steps due to energy constraints of sensor nodes.

In literature, estimation using Petri net models has been studied extensively. For instance, in [23], [24] the authors present an algorithm for obtaining an estimate (and a corresponding error bound) for the state of a given Petri net based on full knowledge of the observed firing sequence but without knowledge of the initial state; these works also discuss how this state estimate may be used to design a controller. In this paper we consider path estimation other than state estimation and our approach is different from those mentioned above. The authors of [25] studied the least-cost firing sequence estimation problem in labeled Petri nets with unobservable transitions and proposed an estimation algorithm with complexity that is polynomial in the length of the observed sequence of labels. In this paper we consider ordinary Petri net models rather than labeled Petri net models.

Note that due to the structure of the given net, the particular final state might be reached from the initial state through paths of different lengths. Therefore, in general, we need to consider all states reachable from the initial state in L or less steps in order to capture the actual least-cost paths that could lead us from the initial state to the final state. Crucial to our algorithmic complexity analysis is the fact that the number of states that are consistent with a path of a certain length is upper bounded by a function that is polynomial in this length (this follows from a translation of the result in [26]). Using this observation, we are able to show that our algorithm can obtain the least-cost paths with complexity that is polynomial in length L.

II. Petri Net Notation

In this section, we provide basic definitions and terminology that will be used throughout the paper. More details about Petri nets can be found in [27], [28].

A Petri net structure is a weighted bipartite graph $N = (P,T,A,W)$ where $P = \{p_1, p_2, \ldots, p_n\}$ is a finite set of *places* (drawn as circles), $T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of *transitions* (drawn as bars), $A \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (from places to transitions and from transitions to places), and $W : A \to \{1, 2, 3, \ldots\}$ is the *weight function* on the arcs.

Let b^-_{ij} denote the integer weight of the arc from place p_i to transition t_j, and b^+_{ij} denote the integer weight of the arc from transition t_j to place p_i ($1 \le i \le n$, $1 \le j \le m$). Note that b^-_{ij} (b^+_{ij}) is taken to be zero if there is no arc from place p_i to transition t_j (or vice versa). We define the *input incident matrix* $B^- = [b^-_{ij}]$ (respectively the *output incident matrix* $B^+ = [b^+_{ij}]$) to be the $n \times m$ matrix with b^-_{ij} (respectively b^+_{ij}) at its i^{th} row, j^{th} column position. The *incident matrix* of the Petri net is defined to be $B \equiv B^+ - B^-$.

A *marking* (state) is a vector $M : P \to Z^+_0$ that assigns to each place in the Petri net a nonnegative integer number of tokens (drawn as black dots). We use M_0 to denote the initial marking of the Petri net. A transition t is said to be *enabled* if each of its input places p_{in} has at least $B^-(p_{in}, t)$ tokens. We use $M[t\rangle$ to denote that transition t is enabled at marking M. An enabled transition t may *fire* and if it fires, it removes $B^-(p_{in}, t)$ tokens from each input place p_{in} of t and deposits $B^+(p_{out}, t)$ tokens to each output place p_{out} of t to yield a new marking $M' = M + B(\cdot, t)$, where $B(\cdot, t)$ denotes the column of B that corresponds to transition t. This is also denoted by $M[t\rangle M'$ and we say marking M' is *reachable* from marking M via the firing of transition t.

Let $\sigma = t_{i1} t_{i2} \ldots t_{ik}$ ($t_{ij} \in T$) be a transition firing sequence. We say σ is enabled with respect to M if $M[t_{i1}\rangle M_1 [t_{i2}\rangle \ldots M_{k-1}[t_{ik}\rangle$; this is denoted by $M[\sigma\rangle$. Let $M[\sigma\rangle M'$ denote that M' is reachable via the firing of transition sequence σ from M, and let $\overline{\sigma}(t)$ be the total number of occurrences of transition t in σ. More specifically, $\overline{\sigma} = [\overline{\sigma}(t_1) \ldots \overline{\sigma}(t_m)]^T$ is the *firing vector* that corresponds to σ. Furthermore, we use $|\sigma|$ to denote the number of transitions in sequence σ. A *cost function* $C : T \to Z^+$ assigns to each transition in the net a positive integer cost. Given a transition firing sequence $\sigma = t_{i1} t_{i2} \ldots t_{ik}$, its total cost is given by $C(\sigma) = \sum_{j=1}^{k} C(t_{ij}) = \sum_{j=1}^{m} C(t_j) \cdot \overline{\sigma}(t_j)$.

Definition 1 Given an initial marking M_0, the set of reachable markings with respect to transition firing sequences of length L is given by $Z(L) = \{M' \mid \exists\sigma : M_0[\sigma\rangle M'$ and $|\sigma| = L\}$.

III. Problem Formulation

The problem we deal with in this paper is the following. Consider a WSN that is modeled as a Petri net N with an initial state M_0 and costs associated with each transition (via a cost function C). Given a final state M (i.e., a target state to be reached) and a positive integer number L (i.e., an upper bound on the length of sequences of transitions whose firings can lead us from the initial state to the final state), we aim at finding the transition firing sequence(s) with length less than or equal to L which: (i) is (are) consistent with the structure of the Petri net, (ii) leads (lead) us from the initial state to the final state, and (iii) has (have) the least total cost.

Clearly, the set of least-cost firing sequence(s) $\{\sigma_{min}\}$ is the solution to the following problem:

$$arg \min_{\sigma} C(\sigma) \;\; s.t. \;\; M_0[\sigma\rangle M \;\; \& \;\; |\sigma| \leq L. \qquad (1)$$

Example 1 Consider the Petri net shown in Fig. 1 with places $P = \{p_1, p_2, p_3, p_4\}$; transitions $T = \{t_1, t_2, t_3, t_4, t_5\}$; and transition costs given by $C(t_1) = 1$, $C(t_2) = 4$, $C(t_3) = 2$, $C(t_4) = 3$, and $C(t_5) = 5$. The initial marking is given by $[2\ 0\ 0\ 0]^T$ and the final marking is taken to be $[1\ 0\ 1\ 0]^T$, whereas the upper bound on the length of the transition firing sequences is set to $L = 2$. In other words, we need to find, among all valid transition firing sequences of length equal to or less than 2 that lead us from the initial marking to the final marking, the ones that have the least total cost. It is not hard to show that there are two possible transition firing sequences, t_2 and t_1t_3, with total costs given by 4 and 3 respectively. Therefore, we conclude that the least-cost transition firing sequence (within length 2) that leads us from the initial marking to the final marking is t_1t_3. □

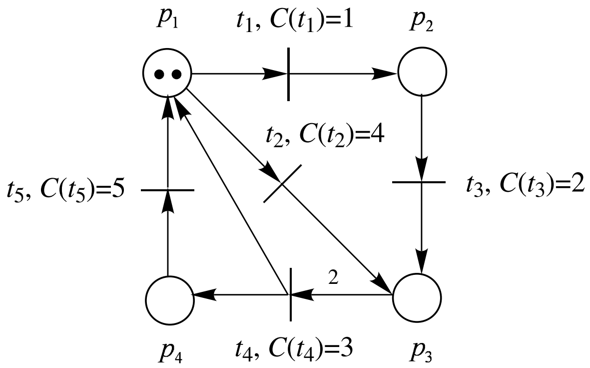

Figure 1. A Petri net with transition costs.

The problem in (1) could be solved by starting from the initial marking, enumerating all transition firing sequences σ with length equal to or less than L, evaluating whether they are enabled or not, if enabled then computing the markings M' resulting from their firings, determining whether they satisfy $M' = M$, and obtaining the valid one(s) with the least cost. The problem with this approach is that, in the worst case, the number of transition sequences considered is exponential in the length of L (e.g., we start with $\sum_{i=1}^{L} m^i$ possible transition firing sequences). However, by looking at this problem in a different way, i.e., in terms of a trellis diagram [29], we will show that one can use a dynamic programming approach to obtain the solution more efficiently in an iterative manner. This approach will take advantage of the fact that several of these sequences visit identical markings at identical points in time (and therefore need not to be explored separately).

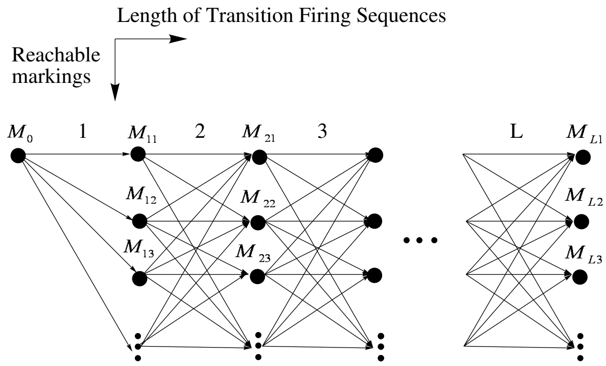

Figure 2. Trellis diagram of the evolution of reachable markings.

In Fig. 2, We depict a trellis diagram that captures the evolution of reachable markings as the length of transition firing sequences increases. In particular, $\{1, 2, \ldots, L\}$ denotes the length of transition firing sequences with time epochs (stages) corresponding to each time a transition fires. Each node in the trellis diagram (drawn as a big black dot) denotes a marking that is reachable from the initial marking through transition firing sequences of length up to the current time epoch, i.e., $M_{ji} \in Z(j)$ where $j \in \{1, 2, \ldots, L\}$ and i is the index of a given marking within the set Z. Arcs between nodes represent transitions whose firings will lead from one marking to another. Given the initial marking M_0, the final marking M, and the upper bound L on the length of transition firing sequences, we need to find the set of transition firing sequences that appear in the trellis diagram and have the least total cost from the initial marking M_0 to the final marking M.

Definition 2 The set of least-cost firing sequences (of length j) that lead to the i^{th} marking M_{ji} at the j^{th} stage of the trellis diagram is given by $LC_i^{(j)} = \{\sigma' \mid \sigma' = arg\min_{\sigma} C(\sigma)$ for σ such that $M_0[\sigma\rangle M_{ji}$ and $|\sigma| = j\}$.

By formulating the problem in terms of a trellis diagram, it is clear that each reachable marking in the set $Z(j+1)$ has to be reached via at least one reachable marking in the set $Z(j)$. Moreover, a dynamic programming approach can be used to compute the least-cost firing sequence(s) iteratively. The basic observation is that the transition firing sequences which have the least cost at time epoch $j+1$ only depend on the least-cost transition firing sequences that reach a marking at time epoch j, and the transitions that can fire at time epoch $j+1$ (from the markings reached at time epoch j). By taking advantage of this observation, one can search for the sequence that has the least cost, one stage in the trellis diagram at a time.

In our setup, at time epoch j, given each reachable marking $M_{ji} \in Z(j)$ and its associated set of least-cost firing sequences $LC_i^{(j)}$, we can compute the set of least-cost firing sequences $LC_i^{(j+1)}$ associated with each reachable marking $M_{(j+1)i} \in Z(j+1)$ at time epoch $j+1$ as follows:

$$LC_i^{(j+1)} = \{\sigma_{j'} t_p \mid [\sigma_{j'}, t_p] = arg \min_{\sigma_{j'},\ t_p} (C(\sigma_{j'}) + C(t_p))\}$$

where $\sigma_{j'}$ and t_p are such that,

$$\exists i' \ s.t.\ \sigma_{j'} \in LC_{i'}^{(j)} \text{ and } M_{ji'}[t_p\rangle M_{(j+1)i}. \tag{2}$$

By calculating (2) iteratively in the length of the transition firing sequences, we can efficiently find the firing sequence(s) that has (have) the least cost to all reachable markings in the trellis diagram. Note that we want to find a particular final marking M that can be reached from the given initial marking M_0. Therefore, once we compute all reachable markings at each stage in the trellis diagram, we need to check whether marking M has appeared or not. If M first appears at a particular stage, the least-cost firing sequence(s) that may lead us from M_0 to M is (are) not necessarily these one(s), because M can potentially be reached from M_0 via a different transition firing sequence that has longer length but smaller cost (M may be reached from the initial marking via transition firing sequences of different lengths). Therefore, we must keep track of all reachable markings within L steps and select, among all sequences whose firings can lead us from M_0 to M, the one(s) that has (have) the least total cost.

Note that each time the final marking M appears in the trellis diagram, we do not need to consider the sequences that emanate from it in the later stage. Since each transition in the net is associated with a positive cost, these sequences emanating from M are guaranteed to have higher cost even if they reach M again.

IV. Obtaining Least-Cost Paths

A. Algorithm

In this section, we propose an iterative algorithm to find the least-cost firing sequence(s), of length less than or equal to the given bound L, that lead from the initial marking M_0 to the final marking M. We use a data structure $\mathscr{C} = (M_{current}, leastcost, (t_{in}, M_{previous}))$ to capture the information we need to store for each node in the trellis diagram. More specifically, at time epoch j: $M_{current}$ denotes the marking that is associated with the node (and can be reached from M_0 through a transition firing sequence of length j); $leastcost$ is the least cost among all valid firing sequences from M_0 to $M_{current}$; (t_{in}, $M_{previous}$) denotes that input transition t_{in} that is enabled at $M_{previous}$ such that the least cost firing sequence goes through $M_{previous}$ at time epoch $j-1$ and leads to $M_{current}$ by firing transition t_{in}.

Note that if M can be reached from M_0 via one or more transition firing sequences, the algorithm outputs the one(s) that has (have) the least total cost; if no such firing sequence exists, the algorithm does not provide any output. We describe the algorithm in details below.

Algorithm 1
Input: A Petri net N with positive costs associated with each transition, a given initial marking M_0, a final marking M, and a maximum length L for transition firing sequences.

1. $\mathscr{C}(0) = \{(M_0,\ 0,\ (\emptyset,\ \emptyset))\}$.
2. Let $j = 1$.
3. Set $\mathscr{C}(j) = \emptyset$.
4. For all $R \in \mathscr{C}(j-1)$ and $R.M_{current} \neq M$ do
    For all t such that $R.M_{current}[t\rangle$
        compute $M' = R.M_{current} + B(\cdot, t)$
        If M' is a new marking that has not appeared in $\mathscr{C}(j)$
            $\mathscr{C}(j) = \mathscr{C}(j) \cup \{(M', R.leastcost + C(t), (t, R.M_{current})\}$
        Else
            M' has appeared in $R' \in \mathscr{C}(j)$
            If $R.leastcost + C(t) < R'.leastcost$
                $R' = (M', R.leastcost + C(t), (t, R.M_{current}))$
            Else If $R.leastcost + C(t) = R'.leastcost$
                $R' = (M', R'.leastcost, R'.(t_{in}, M_{current}) \cup (t, R.M_{previous}))$
            End IF
        End IF
    End For
End For
5. $j = j+1$.
6. If $j = L+1$, Goto 7; else Goto 3.
7. Recover all firing sequences from M_0 to M using the information stored and output the one(s) that has (have) the least total cost. □

Given the Petri net structure its initial marking M_0, Algorithm 1 iteratively computes markings that are reachable from the initial marking along with the least cost of transition sequences that lead from the initial marking to each of these markings. At each time epoch j, the algorithm looks at the set of all enabled transitions and associates each new marking with the transition whose firing (from a reachable marking at stage $j-1$) leads to an overall transition firing sequence that reaches that marking with least total cost. The algorithm then stores all reachable markings and their corresponding least-cost transition(s), and goes to the next step (where the length of the transition firing sequence is increased by one). After considering all valid transition firing sequences with length equal to or less than L (i.e., after we complete L stages of the trellis diagram), the algorithm recovers all transition firing sequences from the initial marking M_0 to the final marking M, and outputs the one(s) that has (have) the least total cost.

Example 2 Recall the Petri net with transition costs shown in Fig. 1. Given the final marking as $M =$

$[1\ 0\ 1\ 0]^T$, we want to find the least-cost transition firing sequences within length 2 (i.e., $L = 2$) that could lead us from the initial marking to this marking. After running Algorithm 1 on this example, the corresponding trellis diagram is shown in Fig. 3. The set of firing sequences that can lead us from M_0 to M is given by $\sigma = \{t_2, t_1t_3\}$. The least-cost firing sequence is $\sigma_{min} = t_1t_3$ and has cost 3. Note that although M_{12} and M_{21} have the same marking, they correspond to different data structures in our algorithm (because they are reached via sequences of different length): one is given by $([1\ 0\ 1\ 0]^T, 4, (t_2, [2\ 0\ 0\ 0]^T))$ and the other by $([1\ 0\ 1\ 0]^T, 3, (t_3, [1\ 1\ 0\ 0]^T))$. Also, after the final marking is reached at M_{12}, we do not consider transitions emanating from it because the sequence will have higher cost when it reaches the final marking again. $\square$

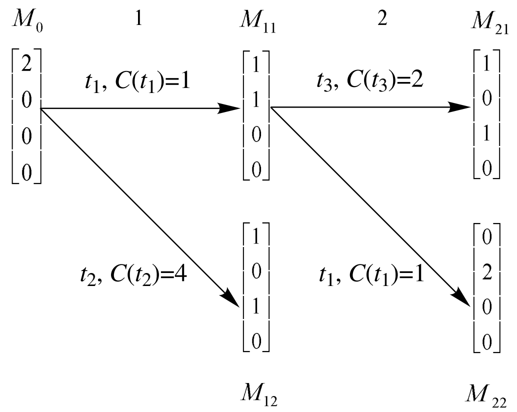

Figure 3. Trellis diagram after running Algorithm 1 for Example 1.

B. Complexity Analysis

Before discussing the complexity of our algorithm, we would like to first translate our problem into the context of labeled Petri nets; then we will use results established in [26] for labeled Petri nets to analyze the complexity of Algorithm 1.

Given the (unlabeled) Petri net structure and its initial marking, we can think of it as a labeled Petri net with all transitions in the net associated with a same label l through a degenerate labeling function E defined below.

A *labeling function* $E : T \to \Sigma$ assigns to each transition in the net a label from a given alphabet Σ. In our setup, we assume that there is only one label $l \in \Sigma$. Therefore, given a transition firing sequence $\sigma = t_{i1}t_{i2}\dots t_{iL}$ of length L, the corresponding label sequence is given by $\omega = E(\sigma) = E(t_{i1})\cdots E(t_{iL}) = \underbrace{l\cdots l}_{L} = l^L$, i.e., a string of length L.

By thinking of the problem in this way, the upper bound L on the number of transitions in a particular transition sequence translates to an upper bound on the number of labels in its corresponding label sequence. Therefore, the problem can be represented equivalently as follows: Consider a labeled Petri net N with initial marking M_0, positive costs associated with each transition (via a cost function C), and a single label l associated with all transitions; given an observed label sequence $\omega = l\cdots l$ of length L that has been generated by an underlying (unknown) firing sequence $t_{i1}t_{i2}\dots t_{iL}$ and a final marking M, find the transition firing sequence(s) that: (i) is (are) consistent with the structure of the net, (ii) leads (lead) us from the initial marking M_0 to the marking M, (iii) has (have) length less than or equal to L, and (iv) has (have) the least total cost.

It was shown in [26] that given an observed sequence of labels of length L, the total number of markings at the L^{th} stage in the trellis diagram is upper bounded by a polynomial function in L, namely L^b where b is a constant associated with structural parameters of the given Petri net[1]. We use this fact in our algorithmic complexity analysis below.

The complexity of Algorithm 1 can be obtained as follows. First, regarding space complexity, the storage needed is proportional to the number of reachable markings (nodes). For each reachable marking (apart from the marking information itself and the least cost to get to it), we need to store the valid pairs of transitions and reachable markings in the previous time epoch that could lead to this marking and have the least cost. Since there are total m transitions in the net, the number of transitions that could lead to the current (reachable) marking from distinct (reachable) markings in the previous stage is bounded by m. Thus, for each reachable marking, the information to be stored is proportional to m. Clearly, since the number of reachable markings at the L^{th} stage in the trellis diagram is upper bounded by L^b, the total space needed to store all reachable markings at every stage in the trellis diagram is $\sum_{j=1}^{L} O(m \cdot j^b)$ which can be simplified as $O(mL \cdot L^b) = O(mL^{b+1})$, i.e., the storage required is polynomial in the length L of the transition firing sequences.

We now proceed to analyze computational complexity. We use n_{j-1} to denote the number of reachable markings at the $(j-1)^{st}$ stage of the trellis diagram ($n_{j-1} = O((j-1)^b)$ and n_j to denote the number of reachable markings at the j^{th} stage of the trellis diagram ($n_j = O(j^b)$). Clearly, at time epoch j, the number of possible transitions enabled from the $(j-1)^{st}$ stage is upper bounded by $n_{j-1} \cdot m$ where m is the number of transitions in the net. For each marking (in the j^{th} stage) yielding from these transitions, we need at most n_j comparisons to decide if it has appeared or not (by searching through the set of existing markings) and at most n_j comparisons to decide if it is equal to the final marking M. Thus, the computational complexity for finding the sequence that has least-cost for all markings at the j^{th} stage is bounded by $n_{j-1} \cdot m \cdot 2n_j$, which has complexity $O(m \cdot j^{2b})$. Thus, for reachable markings over all stages, the total

[1] More specifically, in [26] it is argued that $b = c(d-1)$ where c is the number of nondeterministic labels in the net and d is the maximum number of transitions corresponding to a label in the net. We say a label is nondeterministic if there is more than one transition corresponding to the label. In our setup, $c = 1$ and $d = m$, thus $b = m-1$.

computational complexity is given by $\sum_{j=1}^{L} O(m \cdot j^{2b})$, which can be simplified as $O(mL^{2b+1})$. Therefore, the algorithm has complexity that is polynomial in the length of the transition firing sequences.

V. An Illustrative Example

In this section, we illustrate the algorithm via an example of a WSN. Consider a WSN that is modeled by the Petri net shown in Fig. 4. Initially, six pieces of information are available (modeled by places p_1 to p_6 where each place has a token). Following a number of different ways of transmission, the information transmission is completed (modeled by p_{12}). Therefore, the final state is given by $M = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1]^T$). The Petri net has 12 places $P = \{p_1, p_2, \ldots, p_{12}\}$, 8 transitions $T = \{t_1, t_2, \ldots, t_8\}$ and the initial state $M_0 = [1\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0]^T$. We assume that the cost of each transition is given by the cost vector $C = [5\ 5\ 30\ 20\ 20\ 20\ 30\ 10]^T$. Our goal is to estimate the least-cost transition firing sequence(s) that leads (lead) us from the initial state to the final state under different length constraints.

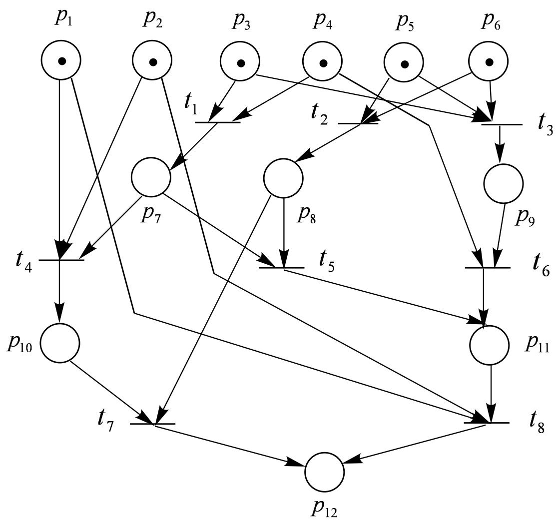

Figure 4. A Petri net model of a WSN.

Assume that we want to find the least-cost transition firing sequences from M_0 to M within length 4. In this case, Algorithm 1 finds the set of least-cost firing sequences to be $\sigma_{min} = \{t_1t_2t_5t_8, t_2t_1t_5t_8\}$ with total cost 40. We provide the following table where *Length* denotes the allowable maximum length of the transition firing sequences considered so far, *Leastcost* captures the least total cost of sequence(s) that satisfies (satisfy) the maximum length constraint and leads (lead) to M from M_0 (we use ∞ to denote that M cannot be reached via transition sequences of the given maximum length from M_0), and $\{\sigma_{min}\}$ is the set of least-cost transition sequence(s) that has (have) least total cost and leads (lead) from M_0 to M.

Length	*Leastcost*	$\{\sigma_{min}\}$
1	∞	$\emptyset$
2	∞	$\emptyset$
3	60	$t_3t_6t_8$
4	40	$t_1t_2t_5t_8$ $t_2t_1t_5t_8$

Remark 1 Note that the final marking is reached via transition firing sequence $t_3t_6t_8$ of length 3 with least cost 60; however, the final marking can also be reached via transition firing sequences of longer length with smaller cost (both $t_1t_2t_5t_8$ and $t_2t_1t_5t_8$ have length 4 with least cost 40). Therefore, in order to capture the least-cost transition firing sequences of length less than or equal to L, we need to consider all markings reached from M_0 within L stages of the trellis diagram.

VI. Conclusions and Future Work

In this paper, we consider the problem of finding the least-cost paths in a WSN that is modeled by a Petri net. In particular, we consider a setting where we are given an initial state and a final state along with a positive integer L that serves as an upper bound on the length of paths that lead us from the initial state to the final state. We assume that each transition in the given net is associated with a positive cost (which could represent its likelihood of occurrence by considering different constraints) and we aim at finding the paths (of length less than or equal to L) which: (i) are consistent with the structure of the Petri net, (ii) lead us from the initial state to the final state, and (iii) have the least total cost. We develop an iterative algorithm that obtains the least-cost paths with complexity that is polynomial in L. This algorithm can be used for least-cost path estimation in WSNs that are modeled as Petri nets.

One direction for future work is to study the timed Petri net model to incorporate temporal information of sensor transmissions. Another interesting extension is to incorporate the constraint of energy consumption of sensor nodes in our Petri net model to optimize the performance of WSNs.

References

[1] C.-Y. Chong and S. P. Kumar, “Sensor networks: evolution, opportunities, and challenges,” *Proc. of the IEEE*, vol. 91, no. 8, pp. 1247–1256, August 2003.

[2] F. Batool and S. Khan, “Traffic estimation and real time prediction using adhoc networks,” in *Proc. IEEE Symp. Emerging Technologies,* pp. 264–269, September 2005.

[3] S. Biswas, R. Tatchikou, and F. Dion, “Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety,” *IEEE Communications Magazine,* vol. 44, no. 1, pp. 74–82, January 2006.

[4] M. Bhardwaj, T. Garnett, and A. Chandrakasan, “Upper bounds on the lifetime of sensor networks,” in *Proc. IEEE Intl. Conf. Communications,* vol. 3, pp. 785–790, June 2001.

[5] C. Jaikaeo, C. Srisathapornphat, and C.-C. Shen, "Diagnosis of sensor networks," in *Proc. IEEE Intl. Conf. Communications,* vol. 5, pp. 1627–1632, June 2001.
[6] B. Warneke, M. Last, B. Liebowitz, and K. Pister, "Smart dust: communicating with a cubic-millimeter," *Computer,* vol. 34, no. 1, pp. 44–51, January 2001.
[7] L. Yin, T. Hong, C. Liu, Y. Xia, and H. Zhou, "A reprogramming protocol based on state machine for wireless sensor network," in *Proc. Intl. Conf. Electrical and Control Engineering,* pp. 232–235, December 2010.
[8] O. Kasten and K. Romer, "Beyond event handlers: programming wireless sensors with attributed state machines," in *Proc. 4th Intl. Symp. Information Processing in Sensor Networks,* pp. 45–52, April 2005.
[9] X. Ye and J. Li, "An FSM-based automatic detection in AODV for ad hoc network," in *Proc. Intl. Symp. Computer Network and Multimedia Technology,* pp. 1–5, December 2009.
[10] S. Kokalj-Filipovic, P. Spasojevic, and R. Yates, "Geographic data propagation in location-unaware wireless sensor networks: a two-dimensional random walk analysis," *IEEE Journal on Selected Areas in Communications,* vol. 27, no. 7, pp. 1158–1168, September 2009.
[11] C. Chi-Kin, Q. Fei, S. Sayed, M. H. Wahab, and Y. Yang, "Harnessing battery recovery effect in wireless sensor networks: Experiments and analysis," *IEEE Journal on Selected Areas in Communications,* vol. 28, no. 7, pp. 1222–1232, September 2010.
[12] B. Bostan-Korpeoglu, A. Yazici, I. Korpeoglu, and R. George, "A new approach for information processing in wireless sensor network database applications," in *Proc. 22nd Intl. Conf. Data Engineering Workshops*, April 2006.
[13] D. He, L. Cui, H. Huang, and M. Ma, "Design and verification of enhanced secure localization scheme in wireless sensor networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 20, no. 7, pp. 1050–1058, July 2009.
[14] A. Shareef and Y. Zhu, "Energy modeling of processors in wireless sensor networks based on Petri nets," in *Proc. 37th Intl. Conf. Parallel Processing*, pp. 129–134, September 2008.
[15] A. Shareef and Y. Zhu, "Energy modeling of wireless sensor nodes based on Petri nets," in *Proc. 39th Intl. Conf. Parallel Processing*, pp. 101–110, September 2010.
[16] C.-H. Kuo and J.-W. Siao, "Petri net based reconfigurable wireless sensor networks for intelligent monitoring systems," in *Proc. Intl. Conf. Computational Science and Engineering*, pp. 897–902, August 2009.
[17] C.-H. Kuo and T.-S. Chen, "PN-WSNA: An approach for reconfigurable cognitive sensor network implementations," *IEEE Sensors Journal*, vol. 11, no. 2, pp. 319–334, February 2011.
[18] S. Yang, H. Cheng, and F. Wang, "Genetic algorithms with immigrants and memory schemes for dynamic shortest path routing problems in mobile ad hoc networks," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 40, no. 1, pp. 52–63, January 2010.
[19] T. Qiu, F. Xia, L. Feng, G. Wu, and B. Jin, "Queueing theory-based path delay analysis of wireless sensor networks," *Advances in Electrical and Computer Engineering*, vol. 11, no. 2, pp. 3–8, March 2011.
[20] Z. Duan, F. Guo, M. Deng, and M. Yu, "Shortest path routing protocol for multi-layer mobile wireless sensor networks," in *Proc. IEEE Intl. Conf. Networks Security, Wireless Communications and Trusted Computing*, pp. 106–110, April 2009.
[21] Y.-L. Lai and J.-R. Jiang, "Optimal multipath planning for intrusion detection in smart homes using wireless sensor and actor networks," in *Proc. 39th IEEE Intl. Conf. Parallel Processing Woshshops,* pp. 562–570, September 2010.
[22] Z. Yu, X. Fu, Y. Cai, and M. C. Vuran, "A reliable energy-efficient multi-level routing algorithm for wireless sensor networks using fuzzy Petri nets," *Journal of Sensors*, vol. 11, no. 3, pp. 3381–3400, March 2011.
[23] A. Giua, "Petri net state estimators based on event observation," in *Proc. 36th IEEE Intl. Conf. Decision and Control*, pp. 4086–4091, December 1997.
[24] A. Giua and C. Seatzu, "Observability of place/transition nets," *IEEE Trans. on Automatic Control*, vol. 47, no. 9, pp. 1424–1437, September 2002.
[25] L. Li and C. N. Hadjicostis, "Least-cost transition firing sequence estimation in labeled Petri nets with unobservable transitions," *IEEE Trans. on Automation Science and Engineering,* vol. 8, no. 2, pp. 394–403, April 2011.
[26] Y. Ru and C. N. Hadjicostis, "Bounds on the number of markings consistent with label observations in Petri nets," *IEEE Trans. on Automation Science and Engineering*, vol. 6, no. 2, pp. 334–344, April 2009.
[27] T. Murata, "Petri nets: properties, analysis and applications," *Proceedings of the IEEE,* vol. 77, no. 4, pp. 541–580, April 1989.
[28] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems,* Springer, 1999.
[29] S. Lin, T. Kasami, T. Fujiwara, and M. Fossorier, *Trellises and Trellis-Based Decoding Algorithms for Linear Block Codes,* Kluwer Academic Publishers, 1998.

Lingxi Li received the B.E. degree in Automation from Tsinghua University, China, in 2000, the M.S. degree in Control Theory and Control Engineering from the Institute of Automation, Chinese Academy of Sciences, China, in 2003, and the Ph.D. degree in Electrical and Computer Engineering from University of Illinois at Urbana-Champaign, USA, in 2008.

In 2008 he joined the Faculty at Indiana University-Purdue University Indianapolis (IUPUI) where he is currently an Assistant Professor with the Department of Electrical and Computer Engineering. His research interests include modeling, diagnosis, and control of complex systems; fault-tolerant systems; discrete event systems; graph theory; with applications to intelligent transportation systems, power systems, communication networks, and biological systems.

Dongsoo Stephen Kim received the M.S. degree in Computer Science from the University of Texas at Dallas, TX, USA, in 1994, and the Ph.D. degree in Computer Science and Engineering from the University of Minnesota, Minneapolis, MN, USA, in 1998. Dr. Kim worked as a research scientist for Electronics and Telecommunications Research Institute from 1986 to 1992, and as a project manager for Megaxess Inc. from 1998 to 2000. In 2000, he joined the Department of Electrical and Computer Engineering, Purdue School of Engineering and Technology, Indiana Univ. Purdue Univ. Indianapolis, Indiana, USA. He is currently an Associate Professor of the Department, Indiana University Purdue University Indianapolis.

His research includes switch networks, optical switches, network survivability, protection switching, network planning, quality-of-service provisioning in the Internet, mobile ad-hoc networks, mobility modeling, sensor networks, and power-aware routing.