Int. J. Electron. Commun. (AEÜ) 69 (2015) 1102–1112

Contents lists available at ScienceDirect

International Journal of Electronics and Communications (AEÜ)

journal homepage: www.elsevier.com/locate/aeue

Optimal query assignment for wireless sensor networks

Mihaela Mitici[a,*], Martijn Onderwater[b], Maurits de Graaf[a,c], Jan-Kees van Ommeren[a], Nico van Dijk[a], Jasper Goseling[a], Richard J. Boucherie[a]

[a] Department of Applied Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands
[b] Centrum voor Wiskunde en Informatica, P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands
[c] Thales B.V., P.O. Box 88, 1270 Huizen, The Netherlands

ARTICLE INFO

Article history:
Received 21 December 2012
Accepted 14 April 2015

Keywords:
Wireless sensor networks
Markov Decision Processes
Quality of Service

ABSTRACT

Increasing computing capabilities of modern sensors have enabled current wireless sensor networks to process queries within the network. This complements the traditional features of the sensor networks such as sensing the environment and communicating the data. Query processing, however, poses Quality of Service challenges such as query waiting time and validity (age) of the data. We focus on the processing cost of queries as a trade-off between the time queries wait to be processed and the age of the data provided to the queries. To model this trade-off, we propose a Continuous Time Markov Decision Process which assigns queries either to the sensor network, where queries *wait* to be processed, or to a central database, which provides stored and possibly *outdated* data. To compute an optimal query assignment policy, a Discrete Time Markov Decision Process, shown to be stochastically equivalent to the initial continuous time process, is formulated. A comparative numerical analysis of the performance of the optimal assignment policy and of several heuristics, derived from practice, is performed. This provides a theoretical support for the design and implementation of WSN applications, while ensuring a close-to-optimum performance of the system.

1. Introduction

Wireless sensor networks (WSNs) are commonly used to sense environmental attributes such as indoor or outdoor level of CO_2, temperature, noise level [1]. The sensed data is stored in databases, from which queries can be processed at a later stage.

The increasing computing capabilities of modern sensors have enabled the WSNs to become an integrated platform on which local query processing is performed. Consequently, not only storage facilities, such as central databases (DB), are able to process queries, but also the sensors within the WSN. Letting the WSN process queries, however, poses Quality of Service (QoS) challenges. For example, sensors can provide queries with the most recently sensed data. But directing the queries always to the WSN can overload the network and lead to high query waiting times. A trade-off arises between processing the queries within the WSN with the most recently acquired data and the time queries wait to be processed.

In recent years, studies on sensor networks have focused mainly on energy efficient data transmission [2–4] and the traffic was assumed to have unconstrained delivery requirements. However, growing interest in applications with specific QoS requirements has created additional challenges. We refer to [2,5] for an extensive outline of WSN specific QoS requirements. The literature reveals related work on QoS-based routing protocols within the sensor network. Most such protocols satisfy end-to-end packet delay [6] or data reliability requirements [7,8] or a trade-off between the two [9]. However, little work exists on QoS guarantees in the field of sensor query monitoring, as addressed in this paper. In [10] a query optimizer is used to satisfy query delay requirements. In [11] the authors use data validity restrictions to specify how much time is allowed to pass since the last sensor acquisition so that the sensors are not always active and some previously sensed data is used.

This paper analyzes the cost of query processing seen as a trade-off between two QoS requirements commonly encountered in WSNs: the time queries wait to be processed and the age of the data provided to the queries. We consider a system consisting of a central DB and a WSN, both able to solve queries (see Fig. 1).

* Corresponding author. Tel.: +31 619702254.
E-mail addresses: M.A.Mitici@utwente.nl (M. Mitici), M.Onderwater@cwi.nl (M. Onderwater), M.deGraaf@utwente.nl (M. de Graaf), J.C.W.vanOmmeren@utwente.nl (J.-K. van Ommeren), N.M.vanDijk@utwente.nl (N. van Dijk), J.Goseling@utwente.nl (J. Goseling), R.J.Boucherie@utwente.nl (R.J. Boucherie).

http://dx.doi.org/10.1016/j.aeue.2015.04.009
1434-8411/

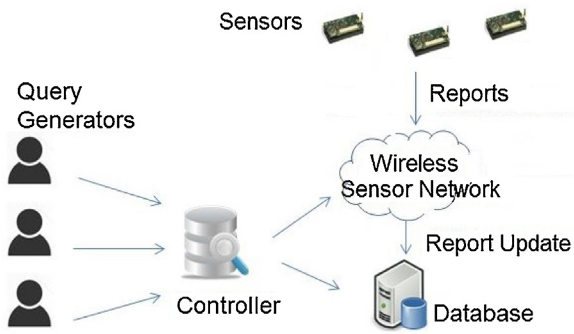


Fig. 1. Wireless sensor network seen as an integrated platform, where queries generated by the end-users are processed either by the WSN or by the DB. Reports are random calls to the WSN and are processed by the WSN. After a report is completed, the DB is updated.

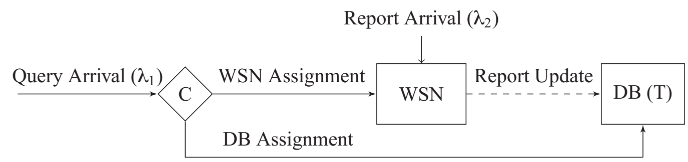


Fig. 2. Proposed model incorporating a controller (C), the database (DB) and the wireless sensor network (WSN). The DB solves queries assigned by the controller. The WSN solves reports and queries assigned by the controller. The data stored in the DB is considered outdated if the age of the data exceeds a validity threshold T. After T is exceeded, the age of the data increases linearly until a report comletion updates again the DB.

We assume that the DB processes queries instantaneously, as the time required to fetch data from the DB is negligible compared to the time a query is processed by the WSN. For the WSN, a processor sharing service type is assumed. This reflects the IEE 802.15.4 MAC design principle of distributing the processing capacity fairly among the jobs simultaneously present in the network. Processor sharing for WLAN was assumed in [11] and validated by simulation in [12].

Queries arriving at a controller are assigned either to the WSN or to the DB. A WSN assignment increases the load of the network and results in large query waiting times. If queries are sent to the DB, the data provided to the queries may be outdated, as the age of the stored data increases over time. The fact that the quality of the stored data deteriorates over time is an essential feature of our system and will pose technical challenges, as seen in the next section. In this paper we provide an approach on how to trade-off between the waiting time of queries and the age of the data, in an optimal and computational manner. In particular, we determine an optimal query assignment strategy that minimizes the processing cost of the queries by trading-off between the waiting time of the queries and the age of the data provided to the queries.

The query assignment problem presented above is formulated as a Continuous Time Markov Decision Process (CTMDP) with a drift. The continuous character of the process, and in particular, the fact that the age component of the process evolves continuously over time, makes the problem non-standard and computationally intractable, i.e. the standard way of deriving an optimal policy recursively using dynamic programming is not directly applicable. Therefore, for computational reasons, we argue a Discrete Time and Discrete State Markov Decision Process. First, we propose a non-standard exponentially uniformized Markov Decision Process, which we show to be *stochastically equivalent* to the original Continuous Time Markov Decision Process with a drift. However, the exponentially uniformized process still contains the age as a continuous state component. Therefore, for further computational tractability, we argue a Discrete Time and Discrete State Markov Decision Process. We then determine an optimal query assignment policy for the discrete time and state process by means of stochastic dynamic programming. Finally, we argue and numerically illustrate that the optimal policy also holds for the original Continuous Time Markov Decision Process with a drift.

In addition, we provide a numerical comparative analysis of the performance of the optimal policy and of several heuristics. The heuristics are simple assignment strategies, commonly used in practice. The results provide a formal support for the design and implementation of query assignment policies in practice so that the system can perform close to the optimum.

The paper is structured as follows. In Section 2, we describe the model of the query assignment problem and define it as a Discrete Time and Space Markov Decision Problem. In Section 4, we assess numerically the performance of our proposed assignment policy and compare it with other feasible heuristics. In Section 5 we discuss the comparative analysis of the proposed assignment policy and the heuristics. Concluding remarks are stated in Section 6.

2. Model formulation

In this section we introduce the query assignment problem formally. In Section 2.2.1, we define the query assignment problem as a Continuous Time Markov Decision Process (CTMDP) with a drift. Next, we construct an exponentially uniformized Markov Decision Process in Section 2.2.2. We show that the exponentially uniformized Markov Decision Process and the initial Continuous Time Markov Decision process (Section 2.2.1) are stochastically equivalent. This leads to the formulation of the query assignment problem as a Discrete Time and Discrete Space Markov Decision Problem in Section 2.2.3.

2.1. Model description

The system consists of a service facility (WSN) with Processor Sharing capabilities and a storage facility (DB). Fig. 2 shows the proposed model.

Two types of jobs: queries and reports, arrive at the system according to a Poisson process. Queries arrive at rate λ_1. Reports arrive at rate λ_2. Reports are requests issued to the WSN to sense the environment and send the data to the DB. Reports update, therefore, the DB. The service requirements of the jobs are exponentially distributed with parameter μ, independently of the job type. To ensure that the system is stable, we assume that $\lambda_2 < \mu$.

Incoming queries are handled by a controller which assigns them either to the DB or to the WSN. When assigned to the DB, queries are immediately answered with stored data. However, the stored data might be outdated, i.e. the age of the data might exceed a validity threshold T. Assigned to the WSN, the queries wait to receive the sensed data, sharing the service with the other jobs present in the network. We assume a Processor Sharing service type of service for the WSN. Therefore, the expected waiting time of the queries is growing linearly with the number of jobs being processed by the WSN (a direct consequence of Little's Law).

The system assumes a query processing cost which is influenced by the type of query assignment (DB or WSN assignment). The cost of a DB assignment is an instantaneous cost, indicating how much the age of the stored data has exceeded a validity threshold T. The cost of a WSN assignment consists of penalties, accumulating over time, related to having queries waiting in the WSN to be processed. These penalties increase upon a WSN assignment, as a consequence of the Processor Sharing service type of the WSN. When a new query arrives at the controller, the model must decide between increasing the query processing cost of the system with an instantaneous DB-related cost or with a WSN-related cost.

Several query assignment heuristics, derived from practice, are analyzed in Section 4. We are interested in finding an optimal assignment strategy and in quantifying the assignment cost of the optimal policy, in comparison to the heuristics. As such, we formulate our problem as a Markov Decision Problem and find an optimal assignment policy that achieves a trade-off between the waiting time of the queries and the age of the data provided to the queries.

2.2. Stochastic dynamic programming formulation

As mentioned earlier, in order to make the query assignment problem computationally tractable, we will follow three steps, provided in Sections 2.2.1, 2.2.2 and 2.2.3.

2.2.1. Continuous Time Markov Decision Process with a Drift

The system presented in Section 2.1 is formally introduced below as a Continuous Time Markov Decision Process with a drift. For an introduction to Continuous Time Markov Decision Processes with a drift, see, for instance [13].

Firstly, at any point in time, the system is completely described by the number of queries, reports and the age of the data stored in the DB. Thus, the state space of the problem is defined as:

- State space $S = \mathbb{N}_0 \times \mathbb{N}_0 \times [0, \infty)$, where $(i, j, t) \in S$ denotes the state with i queries and j reports in the WSN, and the time t since the last report completion (age of the stored data).

Upon a query arrival, the controller assigns the query for processing either to the DB or to the WSN. The action space is, thus, defined as:

- Action: the controller takes an action d from the action space $S_a = \{D, W\}$, where $d = D$ denotes a DB assignment and $d = W$ denotes a WSN assignment.

We define a policy π to be a mapping from the state space $S \to S_a$, which specifies the action $d \in S_a$ the controller takes when the system is in state $(i, j, t) \in S$ and a query arrival occurs. We make the natural assumption that this policy is left-continuous in the age component t, which allows for threshold-type of assignment policies of the form $t > T$, where T is a threshold.

The system has a state transition upon a query arrival, a report arrival, a query completion or a report completion. The rates at which these events happen are as follows:

- The transition rates, when in state $(i, j, t) \in S$ and action $d \in S_a$ is taken:

$$q^d[(i,j,t),(i,j,t)_{\prime}] = \begin{cases} \lambda_1, & (i,j,t)_{\prime} = (i+1,j,t), \quad d = W \\ \lambda_1, & (i,j,t)_{\prime} = (i,j,t), \quad d = D \\ \lambda_2, & (i,j,t)_{\prime} = (i,j+1,t) \\ \mu\phi_1(i,j), & (i,j,t)_{\prime} = (i-1,j,t), i > 0 \\ \mu\phi_2(i,j), & (i,j,t)_{\prime} = (i,j-1,0), j > 0 \end{cases} \tag{1}$$

with $\phi_1(i, j) = i/(i + j)$, $\phi_2(i, j) = j/(i + j)$ indicating the Processor Sharing service discipline assumed for the WSN. The first line of (1) models a query arrival under action $d = W$, i.e. the query is assigned to the WSN for processing. The state space illustrates an increment in the number of queries from i to $i + 1$. The second line of (1) models a query arrival under action $d = D$, i.e. the query is assigned to the DB. In this case, the query is processed immediately, no changes occur in the number of the queries and reports in the system. The third line of (1) models a report arrival. The state of the system illustrates an increment in the number of reports. The fourth line of (1) models a query completion at the Processor Sharing rate $\phi_1(i, j) = i/(i + j)$. The number of queries in the system is decremented to $i - 1$. Lastly, the fifth line of (1) models a report completion at the Processor Sharing rate $\phi_2(i, j) = j/(i + j)$. The age of the stored data is reset to zero as a report is completed and updates the DB with the most recently sensed data.

The above Markov Decision Process has a deterministic drift for the age component, t. This increases linearly as long as no report is completed. Also, we consider two types of decisions. Firstly, the decision to assign an incoming query to the DB affects only the infinitesimal generator of the Continuous Time Markov Decision Process (see second line of (1)). Secondly, the decision to assign a query to the WSN affects both the infinitesimal generator and determines a change in the state of the system (see first line of (1)).

The dynamics of this controlled Markovian decision process are uniquely determined by its infinitesimal generators (see, for instance, [14]). In the case of our system described above, under action d, the generator is specified, for any function $f : S \times S \times (0, \infty) \to \mathbb{R}$, as follows:

$$\mathbf{A}^d f(i,j,t) = \sum_{(i,j,t)_{\prime}} q^d[(i,j,t),(i,j,t)_{\prime}] \cdot f[(i,j,t)_{\prime}] + \frac{d}{dt} f(i,j,t) \tag{2}$$

The generator stated in (2) shows that, over time, two things can happen:1) a jump to a new state $(i, j, t)'$ occurs at rate q^d and the time increases or 2) no jump occurs and the time increases.

The cost of the system is two-fold. Firstly, we consider the cost i of having i queries waiting within the WSN to be processed. This cost gives an overview of the load of the WSN over time. Having a large number of queries in the WSN incurs penalties as the queries need to wait more to be processed. Secondly, we consider an instantaneous cost incurred every time a query is solved by the DB. We incur a penalty for each time unit the age of the stored data exceeds a given threshold T. The two costs illustrate the trade-off between the waiting time of the queries within the WSN and the age of the data provided to the queries. Formally,

- Cost: when in state (i, j, t), a cost rate i for the queries waiting in the WSN and an instantaneous cost $(t - T)^+$, where $x^+ = \max(x, 0)$, upon a DB assignment.

The cost function assumes no explicit communication times. When queries are assigned to the WSN, the communication time is implicitly included in the time the query waits to be processed. In the case of a DB assignment, the processing and communication time are negligible compared to the time a query is processed within the WSN. Therefore, we assume a query is immediately processed when assigned to the DB.

2.2.2. Exponentially uniformized Markov Decision Process

The continuous character of the process described in Section 2.2.1, and in particular, the continuous age component of the process, which evolves over time, make the query assignment problem computationally intractable, i.e. the standard way of deriving an optimal policy recursively using dynamic programming is not applicable for a Continuous Time Markov Decision Process with a drift. More precisely, the method of uniformization, commonly used to make a Continuous Time Markov Decision Process computationally tractable, is not directly applicable due to the drift (the age component evolving over time) of our process. Uniformization, as introduced in [15], is a well-known technique used to transform a continuous time Markov jump process into a discrete time Markov process. When the state is also discrete, the process is referred to as a continuous time Markov chain (see, for instance, [16,17]). In [13,18], time discretization is applied to Continuous Time Markov Decision Processes with a drift component evolving over time. Time discretization is a somewhat similar method to uniformization. Time discretization, however, is an approximative method which leads to technical weak convergence. Moreover, it does not lead to exact computational results, as aimed in this paper. Therefore, to be able to compute an optimal query assignment policy, we construct an exponentially uniformized Markov Decision Process, and show it to be stochastically equivalent to the initial Continuous Time Markov Decision Process with a drift. This implies that the two processes are the same in terms of expected assignment costs and policies. We next construct a Discrete Time and State Markov Decision Process, which is computationally tractable. Based on this process, an optimal assignment policy is computed. We then argue and numerically show that the assignment policy computed also holds for the initial Continuous Time Markov Decision Process with a drift (Section 2.2.1).

We now uniformize the Continuous Time Markov Decision Process with a drift described in Section 2.2.1. First, let B be an arbitrarily large finite number such that $B \geq \lambda_1 + \lambda_2 + \mu$. Next, we construct a process which, at exponential times with parameter B, will have a transition from state $(i, j, t) \in S$, as specified in Section 2.2.1, to $(i, j, t)' \in S$. Denote by s the exponential realization time of this transition. Then, given a transition realization of duration s, the transition probabilities under action $d \in S_a$, from one transition epoch to the next, become:

$$P^d[(i,j,t),(i,j,t)_{\prime}] = \begin{cases} \lambda_1 B^{-1}, & (i,j,t)_{\prime} = (i+1,j,t+s), \quad d = W \\ \lambda_1 B^{-1}, & (i,j,t)_{\prime} = (i,j,t+s), \quad d = D \\ \lambda_2 B^{-1}, & (i,j,t)_{\prime} = (i,j+1,t+s) \\ \mu B^{-1}\phi_1(i,j), & (i,j,t)_{\prime} = (i-1,j,t+s), i > 0 \\ \mu B^{-1}\phi_2(i,j), & (i,j,t)_{\prime} = (i,j-1,0), j > 0 \\ 1-(\lambda_1+\lambda_2+\mu \mathbf{1}_{i+j>0})B^{-1}, & (i,j,t)_{\prime} = (i,j,t+s) \\ 0, & \text{otherwise} \end{cases}$$

Theorem 1. *For any policy π, the exponentially uniformized Markov Decision Process and the original Continuous Time Markov Decision Process with a drift are stochastically equivalent.*

Proof. Appendix A □

A consequence of Theorem 1 is that the expected assignment cost for the exponentially uniformized MDP and the CTMDP with a drift are the same. This, in turn, leads to the same optimal policy for the two processes

Now observe that in the CTMDP with a drift, the actions are only taken upon query arrivals, which occur at exponential times. In the case of the exponentially uniformized MDP, the exponential times have parameter B. Thus, the actions will still be taken at exponential times with parameter B, upon a query arrival. Therefore, it is sufficient to keep track of the number of exponential phases N (Erlang distribution with parameter B and N phases). This allows us to restrict ourselves to a Discrete Time and Space Markov Decision Process in Section 2.2.3. A discrete time and space MDP enables us to compute an optimal assignment policy in Section 4.

In the next Section, therefore, we restrict ourselves to a Discrete Space and Time Markov Decision Problem, with $S = \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0$, where $(i, j, N) \in S$ denotes the state in which there are i queries, j reports and N steps since last report completion, i.e. the age of the data is given by the number of exponential phases N.

2.2.3. Discrete Time and Space Markov Decision Problem

Based on the exponentially uniformized model in Section 2.2.2, we formulate our assignment problem as a Discrete Time and Space Markov Decision Problem (DTMDP) as follows:

- State space: $S = \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0$, where $(i, j, N) \in S$ denotes the state with i queries and j reports in the WSN and N is the age of the stored data, with N the number of steps (exponentially distributed with uniformization parameter B) since the last report completion.
- Action space: Upon a query arrival, the controller takes an action d from the action space $S_a = \{D, W\}$, where $d = D$ is a DB assignment and $d = W$ is a WSN assignment.
- Transition probabilities, when the system is in state $(i, j, N) \in S$ and action $d \in S_a$ is taken:

$$P^d[(i,j,N),(i,j,N)_{\prime}] = \begin{cases} \lambda_{1\prime}, & (i,j,N)_{\prime} = (i+1,j,N+1), \quad d = W \\ \lambda_{1\prime}, & (i,j,N)_{\prime} = (i,j,N+1), \quad d = D \\ \lambda_{2\prime}, & (i,j,N)_{\prime} = (i,j+1,N+1) \\ \mu_{\prime}\phi_1(i,j), & (i,j,N)_{\prime} = (i-1,j,N+1), i > 0 \\ \mu_{\prime}\phi_2(i,j), & (i,j,N)_{\prime} = (i,j-1,0), j > 0 \\ 1-(\lambda_{1\prime}+\lambda_{2\prime}+\mu_{\prime}\mathbf{1}_{i+j>0}), & (i,j,N)_{\prime} = (i,j,N+1) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

with $\phi_1(i, j) = i/(i+j)$, $\phi_2(i, j) = j/(i+j)$ and $\lambda_i' = \lambda_i B^{-1}$, $i \in \{1, 2\}$ and $\mu' = \mu B^{-1}$ as per uniformization (see Section 2.2.2). The first two lines of (3) model query arrivals under action d. The third line of (3) models report arrivals. The fourth and fifth lines of (3) model query and report completions, respectively. The sixth line of (3) is a dummy transition as a result of the uniformization. The last line of (3) prohibits any other state transition. Note that every step, the age is incremented, except when a report is completed, i.e. age reset to zero.

- Cost function: The cost of the system is two-fold. Firstly, when i queries are waiting to be solved within the WSN, the system incurs a cost per unit of time:

$$i \tag{4}$$

This can be interpreted as, each unit of time, the system pays one unit for each waiting query. At the end of a query's service, the system had payed one unit for each unit of time the query was in the system, i.e. the query waiting time. Secondly, if an incoming query is assigned to the DB, an instantaneous penalty is incurred for exceeding the validity threshold T of the stored data:

$$\max(N\prime - T)^+, \quad (x)^+ = \max\{0, x\}. \tag{5}$$

where $N' = N/B$ is the age of the data in time units, i.e. the number of uniformization steps multiplied by the expected length of a step. In this case, the system pays for the time the validity threshold T is exceeded. Considering the cost of having queries waiting in the WSN (4) and the instantaneous cost associated with a DB assignment (5), when the system is in state (i, j, N), the cost incurred per unit of time is:

$$C(i, j, N) = i + \lambda_1 (N\prime - T)^+ \mathbf{1}_{(d=D)}, \quad \text{where}(x)^+ = \max\{0, x\}. \tag{6}$$

Eq. (6) shows that the model assesses the trade-off between increasing the processing cost of the system by the instantaneous cost $(N' - T)$ or by accumulating i units of penalties every time slot, until a change in the number of queries occurs.

Remark: The number of exponential phases approximates the time until a report completion by $t + s = (N + 1) \cdot B^{-1}$. Also, the variance of an Erlang distribution with N phases and parameter B, which is the case for our discretized age, is $(N + 1)/B^2$. As $B \geq \lambda_1 + \lambda_2 + \mu$ can be chosen arbitrarily large (see [19]), by the law of large numbers, for very large B, the distribution of Erlang(N+1,B) will concentrate around $(N + 1) \cdot B^{-1}$. Thus, for large uniformization parameter B, the Discrete Time and State MDP approximates the uniformized MDP arbitrarily close.

The value of the uniformization parameter $B \leq \lambda_1 + \lambda_2 + \mu$ can be seen as a scaling factor that does not influence the results. Several examples have been investigated in Appendix C, also showing no effect of B on the assignment policy. One could expect that, for small values of B, a minor effect on the policy might be present due to the approximation of the age component $N' = N/B$ (see (5)). However, we have not been able to find any such example. In other words, the approach followed is strongly supported, both theoretically and numerically.

Now, the quadruple (S, D, P, C) completely describes the Discrete Time and State MDP. To determine an optimal assignment policy and to use standard dynamic programming, we define the following value function:

$$\mathbf{V}_n(i, j, N) := \text{minimal expected assignment cost over } n \text{ steps starting in state}(i, j, N).$$

Then $\mathbf{V}_n(i, j, N)$ is computed recursively by means of the value iteration algorithm (see, for instance, [20] Section 8.5.1) as follows. First, we consider $\mathbf{V}_0(i, j, N) = 0$. Next, we iterate according to the value iteration algorithm and the following backward recursive equation:

$$\mathbf{V}_{n+1}(i, j, N) = \begin{cases} i\prime + \lambda_1\prime \min \begin{cases} V_n(i+1, j, N+1) \\ (N - T\prime)^+ + V_n(i, j, N+1) \end{cases} \\ +\lambda_2\prime V_n(i, j+1, N+1) \\ +\mu\prime \phi_1(i, j) V_n(i-1, j, N+1) \mathbf{1}_{i>0} \\ +\mu\prime \phi_2(i, j) V_n(i, j-1, 0) \mathbf{1}_{j>0} \\ +[1 - (\lambda_1\prime + \lambda_2\prime + \mu\prime \mathbf{1}_{i+j>0})] V_n(i, j, N+1). \end{cases} \tag{7}$$

where $i' = i/B$ and $T' = T/B$, following uniformization. The first term of (7) is the cost of having i queries in service and a query assigned to either the WSN or the DB. The next three terms represent the cost incurred by a transition due to a report arrival, a query completion and a report completion, respectively. The final term is the dummy term due to uniformization.

Simultaneously with computing $\mathbf{V}_n(i, j, N)$, the algorithm computes a ϵ-optimal stationary policy π_n which associates an optimizing action with the right-hand side of (7) for any state (i, j, N). Given the assignment policy, it is possible to compute the average assignment cost.

Denote the minimal average assignment cost by g^*. Since the underlying Markov chain is ergodic, g^* is independent of the initial state. We approximate g^* using the following bounds developed in [21]:

$$L_n^* \leq g^* \leq L_n^{**}, \quad \text{where} \quad L_n^* = \min[V_{n+1}(i, j, N) - V_n(i, j, N)],$$
$$L_n^{**} = \max[V_{n+1}(i, j, N) - V_n(i, j, N)]. \tag{8}$$

In (8), L_n^* is the minimum difference of the value function over two iteration steps, n and $n+1$, whereas L_n^{**} is the maximum difference of the value function over steps n and $n+1$. For $n \to \infty$, L_n^* and L_n^{**} become arbitrarily close. The optimal cost g^* is computed with an accuracy ϵ by iterating the right-hand side of (7) for n times until $L_n^{**} - L_n^* \leq \epsilon/B$ with B the uniformization parameter. Then, the average assignment cost is approximated as

$$g^* = \frac{(L_n^{**} + L_n^*)}{2}. \tag{9}$$

It can be shown that the lower and upper bound converge in a finite number of steps (Theorem 8.5.4 [20]) to the ϵ-optimal cost.

Remark: The approach proposed in Section 2.2 for a Continuous Time Markov Decision Process with a drift can be used to determine an optimal policy for any given cost function and for any fixed policy. In this paper, we chose a cost function that reflects the trade-off between the waiting time of queries in the WSN and the age of the data provided to the queries.

3. Query assignment heuristics

In practice, simple assignment heuristics are employed to manage the query traffic. We consider the following three assignment heuristics, derived from practical assignment strategies:

- A fixed heuristic policy π^{DB} that always assigns incoming queries to the DB. Upon a query arrival, the cost incurred is $(N - T)^+$.
- A fixed heuristic policy π^{WSN} that always assigns incoming queries to the WSN.
- A heuristic policy π^T that always assigns incoming queries to the DB if the age does not exceed the validity threshold, i.e. $N \leq T$, and to the WSN otherwise.

In the next section, we will numerically compare the performance of the above heuristics with the performance of the optimal policy, under different parameter assumptions.

The following theorem states a closed-form expression for the expected assignment costs incurred by the π^{DB} and π^{WSN} heuristics when a validity threshold T is assumed.

Theorem 2. *Assuming the DTMDP parameters λ_1', λ_2' and μ', the average assignment cost of the heuristics π^{DB} and π^{WSN} are as follows,*

$$C_{\pi^{DB}} = \frac{\lambda_1\prime (1 - \lambda_2\prime)^{T+1}}{\lambda_2\prime}, \tag{10}$$

$$C_{\pi^{WSN}} = \frac{\lambda_1\prime}{\mu\prime - (\lambda_1\prime + \lambda_2\prime)}. \tag{11}$$

Proof. Appendix B □

4. Numerical results

4.1. Numerical results – optimal query assignment policy

Based on the Discrete Time and State Markov Decision Process defined in Section 2.2.3, we are able to compute an optimal query assignment policy.

Fig. 3 shows which action is optimal when the system is in state (i, j, N) and validity threshold T is assumed. We fix N, the age of the data, and T, and for every grid point (i, j), i.e. i queries and j reports waiting in the WSN, blue indicates that it is optimal to assign an incoming query to the WSN and white to the DB.

We first analyze the optimal policy for a fixed age of the stored data, $N = 30$ (see Fig. 3(a) and (b)). When T is small (Fig. 3(a)), the DB data is outdated most of the time. As a consequence, it is often optimal to send incoming queries to the WSN. If T is increased (Fig. 3(b)), then the DB data is considered valid for a longer time. As a result, a DB assignment becomes more frequently optimal.

Next, we analyze the effect of increasing N on the optimal policy, while T is fixed. Fig. 3(a) and (c) show the optimal policy when $T = 1$. In the case of large N (Fig. 3(c)), assigning a new query to the DB leads to large penalties generated by a more outdated DB data. Thus, a query is more frequently assigned to the WSN. The same behavior is illustrated for $T = 4$ in Fig. 3(b) and (d), where it is shown that a higher proportion of WSN assignments is associated with high data age.

Lastly, the optimal policy has a switching structure, i.e. once the number of jobs in the WSN reaches a certain threshold, the optimal policy starts sending incoming queries to the DB and continues to do so as the load (number of jobs) of the WSN increases. Moreover, the policy seems to be truncated at the boundary, for $j = 0$. This boundary effect is caused by the interaction between the number of reports j and the cost $(N - T)^+$, as indicated in equation (7), first and fourth line. Away from the boundary, the switching structure of the optimal policy is maintained.

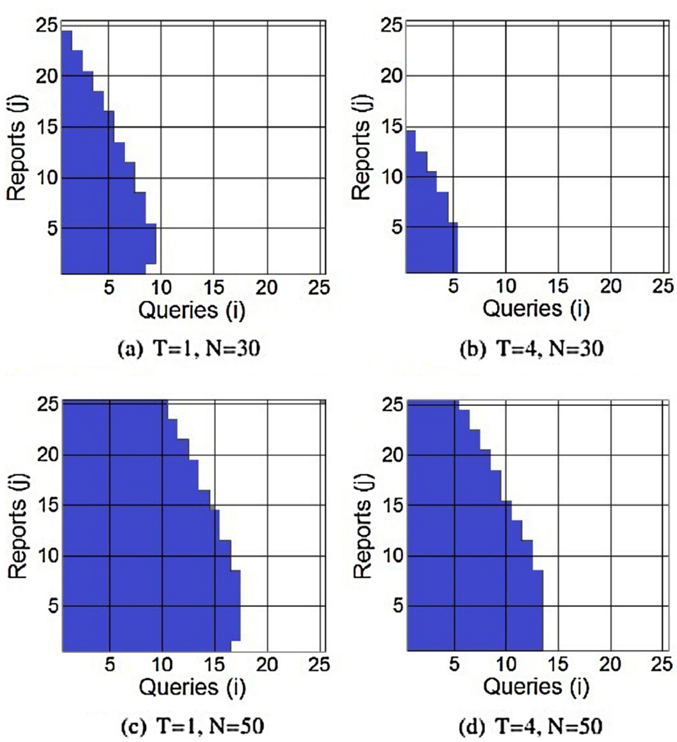


Fig. 3. WSN assignment (dark color) and DB assignment (white) with $\lambda_1 = 0.8$, $\lambda_2 = 0.5$ and $\mu = 1.8$. N is the age of the data and T the validity threshold. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

4.2. Simulation results

We compare the performance of the proposed assignment policy, i.e. the associated average cost (g^*), as defined in equation (9), with the average assignment cost of the heuristics proposed in Section 3 by means of a discrete event simulation. The performance of the assignment policies is investigated under different parameter assumptions. While for π^{DB} and π^{WSN} exact results are presented in Theorem 2, we use simulation to compute the average assignment costs for heuristic π^T. Simulation is also used to determine the fraction of time queries are sent to the DB and WSN. This gives us an indication of the load of the WSN over time.

4.2.1. The influence of the validity threshold on the average assignment cost

Fig. 4 shows that the optimal policy outperforms the heuristics when various validity thresholds T are assumed. Fig. 4(a) shows that, compared with the heuristics, the proposed policy achieves a lower assignment cost. The cost difference is significant for small T. This is of particular interest for real-time applications which are expected to require low validity thresholds.

As expected, at the limit, $T \to \infty$, both π^T and π^{DB} approach the optimal policy. This is explained by the fact that, for large validity thresholds, the stored data is considered valid for a long time. Consequently, it is frequently optimal to send an incoming query to the DB (see Fig. 4(b)). Thus, as T increases, the optimal policy and the heuristic π^T start indicating a DB assignment as the best assignment decision for incoming queries, i.e. both the optimal policy and π^T start behaving as heuristic π^{DB}.

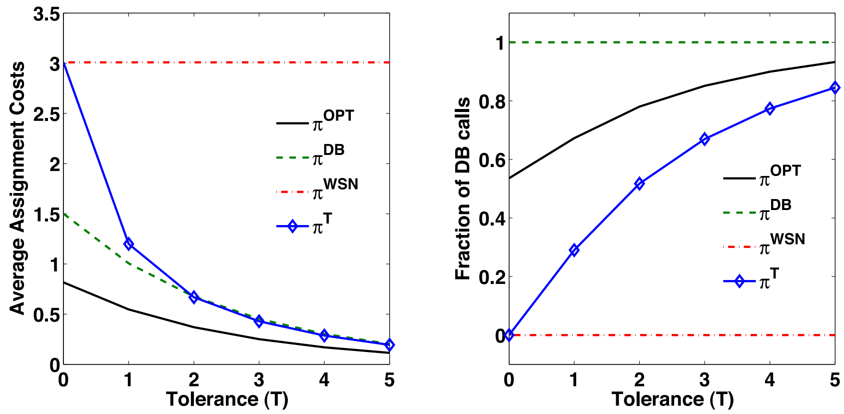
4.2.2. The influence of job arrival rates on the average assignment cost

Fig. 5 shows that the optimal policy outperforms the heuristics when various query arrival rates and validity thresholds are assumed. For stability, $\lambda_2 < \mu$ is assumed (for policy π^{WSN}, $\lambda_1 + \lambda_2 < \mu$). Fig. 5(a) shows that the average assignment cost of π^{WSN}, which is based only on the waiting time of the queries, increases rapidly, as λ_1 increases. A high λ_1 leads to a large number of jobs in the system and, correspondingly, to a high query waiting time. The assignment costs under policy π^{WSN} are independent of T (see Fig. 5(a) and (b)).

The policies π^{DB} and π^T perform closely to the optimal policy, especially for low λ_1. For large T, see Fig. 5(b), the stored data is considered valid for a longer time, leading to lower penalties related to the age of the data and, correspondingly, to lower assignment costs in comparison to the costs in Fig. 5(a). As λ_1 increases, Fig. 5(a) and (b) show a switching point when policy π^T results in higher assignment costs than policy π^{DB}. This shows that when the number of incoming queries is expected to be high, it is preferred, from an assignment cost point of view, to send the queries most of the time to the DB. This enables the WSN to process reports faster so that the DB is updated often and the queries are provided with up-to-date data.

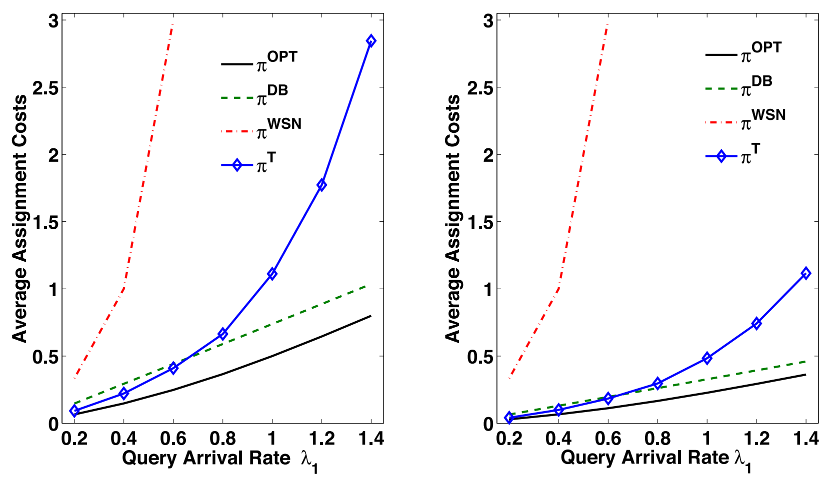
Fig. 6 also shows that the optimal policy outperforms the heuristics when various report arrival rates and validity thresholds are assumed. Again, for stability, $\lambda_2 < \mu$ is assumed (for policy π^{WSN}, $\lambda_1 + \lambda_2 < \mu$). As expected, the assignment costs under policy π^{WSN} increase as λ_2 increases. The assignment cost under π^{WSN} is independent of T, see Fig. 6(a) and (b). Policies π^{DB} and π^T converge to the optimal policy as λ_2 increases.

For large T (see Fig. 6(a) in comparison to Fig. 6(b)), the penalties related to a DB assignment decrease and, consequently, the assignment costs decrease. As λ_2 increases, Fig. 6(a) and (b) show a



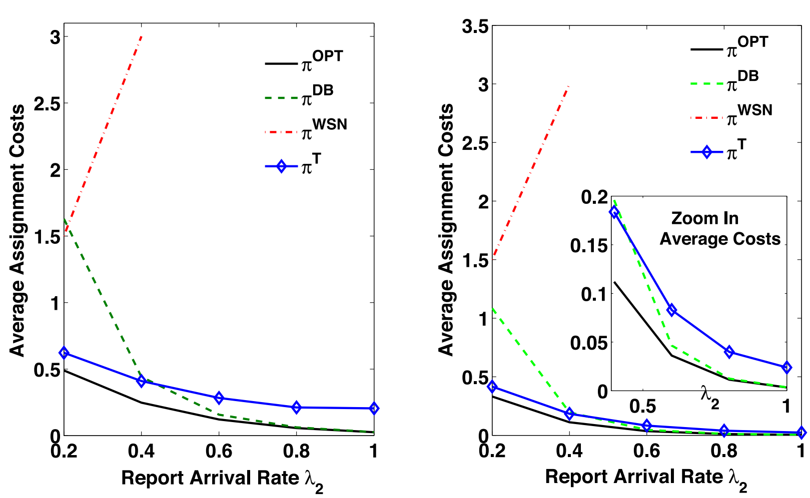
(a) Average Assignment Cost (b) DB utilization

Fig. 4. Average assignment cost and DB Utilization for different validity thresholds T, $\lambda_1 = 0.6$, $\lambda_2 = 0.4$, $\mu = 1.2$.



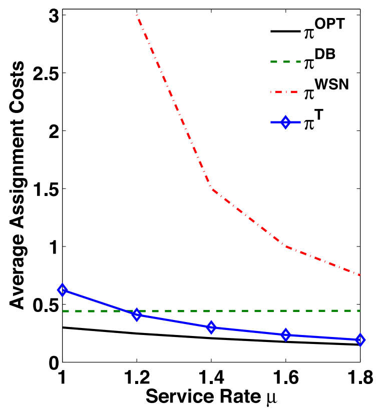
(a) Average Assignment Cost,T = 3. (b) Average Assignment Cost,T = 5.

Fig. 5. Average assignment cost for different query arrival rates, $\lambda_2 = 0.4$, $\mu = 1.2$.

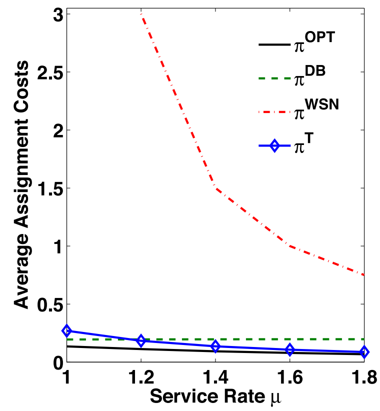


(a) Average Assignment Cost,T = 3. (b) Average Assignment Cost,T = 5.

Fig. 6. Average assignment cost for different report arrival rates $\lambda_1 = 0.6$, $\mu = 1.2$.



(a) Average Assignment Cost, T = 3.



(b) Average Assignment Cost, T = 5.

Fig. 7. Average assignment cost for different processing capabilities μ, $\lambda_1 = 0.6$, $\lambda_2 = 0.4$.

switching point when policy π^{DB} results in lower assignment costs than policy π^{T}. This shows that as the number of expected incoming reports increases, it is preferred, from an assignment cost point of view, to process only reports in the WSN. Consequently, the DB is frequently updated and provides the queries with up-to-date data.

Figs. 5 and 6 quantify the performance of such heuristics, for different arrival rates of the jobs and validity thresholds. Moreover, looking at Figs. 5 and 6, it is possible to optimize parameters such as λ_2 or T so that the heuristics perform close to the optimum. For example, Fig. 6 gives an indication on how large λ_2 should be, when λ_1, μ, T are fixed, such that heuristic π^{DB} performs close to the optimum.

4.2.3. The influence of service rate on the average assignment cost

Fig. 7 shows that in the case of increasing μ, the optimal policy outperforms the heuristics in terms of average assignment costs. For stability, $\lambda_2 < \mu$ (for policy π^{WSN}, $\lambda_1 + \lambda_2 < \mu$).

As expected, Fig. 7(a) shows the assignment cost of heuristic π^{WSN} decreasing as μ increases (the system behaves as an $M/M/1$-Processor Sharing queue with arrival rate $\lambda_1 + \lambda_2$, for which it is known that the expected waiting time of a job is $1/(\mu - \lambda_1 - \lambda_2)$. For large μ, all three heuristics converge to the optimal policy. Heuristic π^{WSN} converges slowly to the optimal policy. The service rate needs to be very high so that assigning all queries to the WSN becomes optimal. Fig. 7(a) and (b) shows a switching point when heuristic π^{T} starts to converge faster to the optimal policy than π^{DB}. This shows that, for large μ, it is not optimal to assign queries only to the *DB*, but rather use the policy π^{T}.

5. Discussion

The comparative analysis of different query assignment policies under various parameter assumptions in Section 4.2 provides insight into the performance of the system. For reasonably large validity thresholds, simple heuristics such as π^{DB} or π^{T} perform well in comparison to the optimal policy. Such heuristics are particularly suitable for monitoring environments with little variation over time, e.g. temperature sensing. However, for applications with highly constrained delivery requirements and large data variance over time, such as fire detection or CO_2 monitoring, T is expected to be low. In this case, our proposed assignment policy shows significant cost savings in comparison to the heuristics.

Moreover, simple heuristics like π^{DB} and π^{T}, which require little computational resources, are preferred for implementation. The risk of implementing them is that they may perform far from the optimum. The comparative analysis of the heuristics and the optimal policy, while considering various system parameters, can provide support for the WSN providers to design their systems in such a way that simple heuristics, which are easy to implement in practice, can perform closely to the optimal policy. For example, while the arrival of queries is generally independent of the WSN service providers, the arrival rate of the reports or the validity threshold can be optimized so that even simple heuristics can have a close-to-optimum performance.

6. Conclusion and future work

We provided a formal support for the analysis of query processing strategies for wireless sensor networks. We determined, using a Markov Decision Processes framework, an optimal assignment policy which assigns queries for processing either to the WSN or to a central DB. The optimal policy is based on the trade-off between the penalties related to having queries waiting to be process in the WSN and an instantaneous cost related to the age of the data stored at a central DB. We assessed numerically the performance of the optimal policy in comparison to several heuristics, commonly used in practice. We showed that the proposed model achieves significant cost savings, especially in the case of real-time applications, where the validity threshold is expected to be low. We compared several heuristics and the optimal policy under different parameter assumptions. The comparative analysis provided a formal support to design query-based systems in such a way that simple heuristics, such as always let the DB process incoming queries, perform close to the optimum. Moreover, this insight can ease the implementation of WSN applications, while ensuring a close-to-optimum performance of the system. Future work includes implementing the proposed model and assessing its performance for real-life WSN applications.

Acknowledgements

This work was performed within the project RRR (Realisation of Reliable and Secure Residential Sensor Platforms) of the Dutch program *IOP Generieke Communicatie*, number IGC1020, supported by the *Subsidieregeling Sterktes in Innovatie*.

Appendix A. Proof of Theorem 1

Proof. Uniformization is commonly used for Markov jump processes, making the problem computationally tractable. As a drift component is introduced in the present setting (the age component of our process evolves continuously over time), this is no longer standard.

The infinitesimal generators uniquely define a Markov process. Therefore, it is sufficient to show that the infinitesimal generators of the exponential uniformized Markov Decision Process and the original Continuous Time Markov Decision Process with a drift are identical.

To prove this, let $P^d_{\Delta t}$ denote the transition probability measures over time interval of length Δt, given that at the last jump the system is in state (i, j, t) and that following a next jump, decision d is taken. As we implicitly made the assumption that a policy π, prescribing an action d upon a query arrival, when the system is in state (i, j, t), is left continuous and since the set of decisions is finite and discrete, for any state (i, j, t) and fixed policy π there exists $\Delta t >= 0$ such that:

$$\pi(i, j, t+s) = \pi(i, j, t) = d, \quad \text{for all } s \le \Delta t.$$

Let $f : \mathbb{N} \times \mathbb{N} \times \mathbb{R}$ be an arbitrary real valued function, differentiable in t and $o(\Delta t)^2 \le Co(\Delta t)^2$ for any constant C. Then by conditioning upon the exponential jump epoch with variable B and for arbitrary f we obtain,

$$\begin{aligned} P^d_{\Delta t} f(i, j, t) &= e^{-\Delta t \cdot B} f(i, j, t+\Delta t) + \int_0^{\Delta t} Be^{-sB} \sum_{(i,j,t)'} P^d[(i, j, t), (i', j', t+s)] f(i', j', t+s) ds + o(\Delta t)^2 \\ &= f(i, j, t+\Delta t) - \Delta t B f(i, j, t+\Delta t) + \Delta t B \sum_{(i',j') \neq (i,j)} q^d[(i, j, t), (i', j', t)] f(i', j', t+\Delta t) B^{-1} \\ &\quad + \Delta t B[1 - q^d(i, j) B^{-1}] f(i, j, t+\Delta t) + o(\Delta t)^2 \\ &= f(i, j, t+\Delta t) + B \sum_{(i',j') \neq (i,j)} q^d[(i, j, t), (i', j', t)][f(i', j', t+\Delta t) - f(i, j, t+\Delta t)] + o(\Delta t)^2, \end{aligned}$$

with $q^d[(i, j, t), (i', j', t)] = q^d[(i, j, t+s), (i', j', t+s)]$ for any '$(i', j') \neq (i, j)$ and arbitrary s. The term $o(\Delta t)^2$ reflects the probability of at least two jumps and the second term of the Taylor expansion for $e^{-\Delta B}$.

Hence, by subtracting $f(i, j, t)$, dividing by Δt and letting $\Delta t \to 0$, we obtain,

$$\begin{aligned} \frac{P^d_{\Delta t} f(i, j, t) - f(i, j, t)}{\Delta t} &= [f(i, j, t+\Delta t) - f(i, j, t)]/\Delta t + B[f(i, j, t+\Delta t) - f(i, j, t)] + o(\Delta t)^2 \\ &\quad + \sum_{(i',j') \neq (i,j)} q^d[(i, j, t), (i', j', t)][f(i', j', t) - f(i, j, t)] \to \frac{d}{dt} f(i, j, t) \\ &\quad + \sum_{(i',j') \neq (i,j)} q^d[(i, j, t), (i', j', t)][f(i', j', t) - f(i, j, t)] = \mathbf{A}^d f(i, j, t) \text{which is the generator in (2).} \end{aligned}$$

Since the exponentially uniformized Markov decision process (as defined in Section 2.2.2) and the Continuous Time Markov Decision Process with a drift (defined in Section 2.2.1) share the same generators (see [14]), the two processes are stochastically equivalent.

□

Appendix B. Proof of Theorem 2

Proof. We first analyze the expected assignment cost under the policy π^{WSN}. The π^{WSN} policy is independent of the validity threshold. The WSN behaves as a M/M/1 Processor Sharing queue. Thus, the cost of the heuristic is given by the expected number of jobs in the WSN as follows,

$$\begin{aligned} C^{WSN}_{\pi} &= \mathbb{E}(i) = \frac{\lambda_{1'}}{\lambda_{1'} + \lambda_{2'}} \cdot \mathbb{E}(i+j) = \frac{\lambda_{1'}}{\lambda_{1'} + \lambda_{2'}} \cdot \frac{\lambda_{1'} + \lambda_{2'}}{\mu - (\lambda_{1'} + \lambda_{2'})} \\ &= \frac{\lambda_{1'}}{\mu' - (\lambda_{1'} + \lambda_{2'})} \end{aligned} \tag{B.1}$$

We now analyze the expected assignment cost under the policy π^{DB}. Under policy π^{DB}, we have a state space with only two components (j, N), where j is the number of reports waiting to be processed and N is the age of the data.

We define the cost of the policy π^{DB} in terms of the limiting probabilities as follows,

$$C_{\pi^{DB}} = \lambda_{1'} \sum_{N \ge T} \pi_{Age}(N) \cdot (N - T)^+, \tag{B.2}$$

where $\pi_{Age}(N) = \sum_{j \ge 0} \pi(j, N)$ is the long-run proportion of time that the process is in state N. In words, $\pi_{Age}(N)$ is the expected number of times the system is in a state with age N during a cycle, divided by the expected length of a cycle, where a cycle is defined as the time between two consecutive resets of the age and no reports in the WSN, i.e. arriving at a state (0, 0).

To compute $C_{\pi^{DB}}$, we need to determine $\pi_{Age}(N) = \sum_{j \ge 0} \pi(j, N)$. The limiting probability $\pi_{Age}(N)$ is the solution of the following balance equations for components j and N:

$$\begin{cases} \pi(j, N) = \lambda_{2'} \pi(j-1, N-1) + (1 - \lambda_{2'} - \mu') \pi(j, N-1), & j \ge 1, N \ge 1 \\ \pi(0, N) = \pi(0, N-1)(1 - \lambda_{2'}), & j = 0, N \ge 1 \\ \pi(j, 0) = \sum_{N \ge 1} \pi(j+1, N) \mu, & j \ge 0. \end{cases} \tag{B.3}$$

The first line of (B.3) shows that we arrive in a state with j reports and age N: (1) from a state where there are $j-1$ reports, the age is $N-1$ and a report arrival occurs or (2) from a state where there are j reports, the age is N and no event occurs. The second line of (B.3) shows that we arrive in a state with zero reports and age N only from a state with zero reports, age is $N-1$ and no report arrival occurs. Lastly, the third line of (B.3) shows that we arrive in a state with age zero and j reports from a state with $j+1$ reports, age $N \ge 1$ and a report completion occurs.

The system (B.3) can be solved after determining the limiting probability $\pi(0, N)$, $N \geq 1$, for component age, and $\pi(j, 0)$, $j \geq 0$, for component reports.

We first determine $\pi(j, 0)$. Let $\pi_{Report}(j) = \sum_{N\geq 0} \pi(j, N)$. Then the balance equations for component j, number of reports, are as follows:

$$\begin{cases} \pi_{Report}(0) = \mu' \pi_{Report}(1) + (1 - \lambda_{2'})\pi_{Report}(0) \\ \pi_{Report}(j-1) = \mu' \pi_{Report}(j) + (1 - \lambda_{2'} - \mu')\pi_{Report}(j-1) \\ \qquad + \lambda_{2'} \pi_{Report}(j-2), j \geq 2 \\ \sum_{j\geq 0} \pi_{Report}(j) = 1. \end{cases} \tag{B.4}$$

$$\begin{cases} \pi_{Age}(N) & = \sum_{j\geq 0} \pi(j, N) \\ & = \sum_{j>0} (1 - \lambda_{2'} - \mu')\pi(j, N-1) + \sum_{j>0} \lambda_{2'} \pi(j-1, N-1) + (1 - \lambda_{2'})\pi(0, N-1) \\ & = \sum_{j>0} (1 - \lambda_{2'} - \mu')\pi(j, N-1) + \sum_{j>0} \lambda_{2'} \pi(j-1, N-1) + (1 - \lambda_{2'} - \mu')\pi(0, N-1) + \mu' \pi(0, N-1) \\ & = \sum_{j>0} (1 - \lambda_{2'} - \mu')\pi(j, N-1) + \sum_{j\geq 0} \lambda_{2'} \pi(j, N-1) + (1 - \lambda_{2'} - \mu')\pi(0, N-1) + \mu' \pi(0, N-1) \\ & = \sum_{j\geq 0} (1 - \lambda_{2'} - \mu')\pi(j, N-1) + \sum_{j\geq 0} \lambda_{2'} \pi(j, N-1) + \mu' \pi(0, N-1) \\ & = \sum_{j\geq 0} (1 - \mu')\pi(j, N-1) + \mu' \pi(0, N-1); \\ \pi_{Age}(0) & = \sum_{j\geq 0} \pi(j, 0) \\ & = \mu' \sum_{j\geq 0} \pi(j+1, N) = \mu' \sum_{j\geq 1} \pi(j, N) = \mu' \sum_{j\geq 0} \pi(j, N) - \mu' \pi(0, N); \\ \sum_{N\geq 0} \pi_{Age}(N) & = 1. \end{cases} \tag{B.11}$$

The first line of (B.4) shows that we arrive in a state with zero reports in two ways: 1) from a state with one report, a report completion occurs and we jump to a state with zero reports or 2) from a state with zero reports and no arrival of a report occurs (no report completion is possible here since there are no reports to be completed), thus, we remain in a state with zero reports.

From (B.4), using substitution, it follows that

$$\pi_{Report}(k) = \left(\frac{\lambda_{2'}}{\mu'}\right)^j \pi_{Report}(0), \quad j > 0. \tag{B.5}$$

Using (B.5) in the last line of equation (B.4),

$$\begin{aligned} & \sum_{j\geq 0} \pi_{Report}(j) = 1 \\ & \sum_{j\geq 0} \pi_{Report}(0) \left(\frac{\lambda_{2'}}{\mu'}\right)^j = 1 \\ & \pi_{Report}(0) = 1 - \frac{\lambda_{2'}}{\mu'} \end{aligned} \tag{B.6}$$

We next determine $\pi(0, 0)$, the long-term proportion of time that the state is in state $(0, 0)$, i.e. zero reports and age zero. Notice that we can arrive at the state $(0, N)$ only if we are in state $(0, N-1)$ and no arrival of a report occurs.

$$\pi(0, N) = (1 - \lambda_{2'})\pi(0, N-1), \quad N \geq 1 \tag{B.7}$$

Solving (B.7), it follows that:

$$\pi(0, N) = (1 - \lambda_{2'})^N \pi(0, 0) \tag{B.8}$$

Now, the proportion of time we are in a state with zero reports is the proportion of time we are in a state where there are zero reports and the age is $N \geq 0$. Thus, using (B.8),

$$\pi_{Report}(0) = \sum_{N\geq 0} \pi(0, N) = \sum_{N\geq 0} (1 - \lambda_{2'})^N \pi(0, 0) = \frac{1}{\lambda_{2'}} \pi(0, 0). \tag{B.9}$$

From (B.6) and (B.9), we obtain:

$$\pi(0, 0) = \frac{(\mu' - \lambda_{2'})\lambda_{2'}}{\mu'} \tag{B.10}$$

We next state the balance equations for the age component N,

The second line of (B.11) shows that we arrive at a state with age N and $j > 0$ reports either 1) from a state with $j > 0$ reports, age $N-1$, and there is no report completion and no report arrival or 2) from a state with $j-1 \geq 0$ reports, age $N-1$ and a report arrival occurs. If the age is N and there are no reports, $j = 0$, we arrive in this state from a state $(0, N-1)$ and no report arrival occurs (here, a report completion is not possible). In equation (B.11), the component $\pi_{Age}(0)$ shows that we arrive at a state with age 0 and j reports after a report completion.

Denoting $\pi_{Age}(N) = \sum_{j\geq 0} \pi(j, N)$, equation (B.11) now becomes:

$$\begin{cases} \pi_{Age}(N) = (1 - \mu')\pi_{Age}(N-1) + \mu' \pi(0, N-1), N \geq 1 \\ \pi_{Age}(0) = \mu' \sum_{N\geq 0} \pi(N) - \mu' \sum_{N\geq 0} \pi(0, N), \text{with} \pi(N) = \sum_{j\geq 0} \pi(j, N) \\ \sum_{k\geq 0} \pi_{Age}(k) = 1. \end{cases} \tag{B.12}$$

Observe that $\sum_{N\geq 0} \pi(N) = 1$ and $\sum_{N\geq 0} \pi(0, N) = \pi_{Report}(0) = \frac{1}{\lambda_{2'}} \pi(0, 0)$ as per (B.9). Using these results in the second line of (B.12), it follows that:

$$\begin{cases} \pi_{Age}(N) = (1 - \mu')\pi_{Age}(N-1) + \mu' \pi(0, N-1), N \geq 1 \\ \pi_{Age}(0) = \mu' [1 - \frac{1}{\lambda_{2'}} \pi(0, 0)] \\ \sum_{N\geq 0} \pi_{Age}(N) = 1 \end{cases} \tag{B.13}$$

We solve (B.13) by induction, using that $\pi(0, N-1) = (1 - \lambda_2')^{N-1} \pi(0, 0)$, as per (B.8), and $\pi(0,$

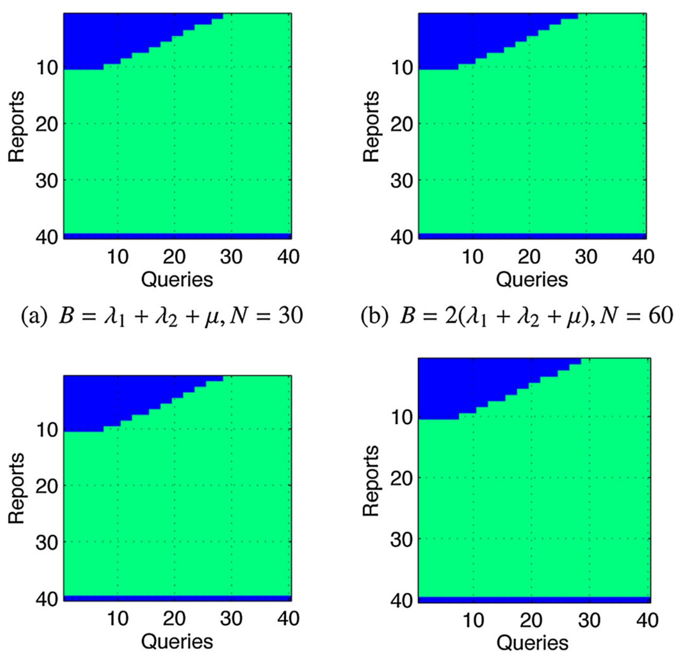
$0) = ((\mu' - \lambda_2{}')\lambda_2{}'/\mu)$, as per (B.10). Solving for (B.13), we have that

$$\pi_{Age}(N) = \lambda_{2'}(1-\lambda_{2'})^N \tag{B.14}$$

Using (B.14), we can now compute the cost (B.2) as follows,

$$\begin{aligned}
C_{\pi^{DB}} &= \lambda_{1'} \sum_{N\geq 0} \pi_N(N)\cdot(N-T)^+ \\
&= \lambda_{1'} \sum_{N'\geq 0} \lambda_{2'}(1-\lambda_{2'})^{N'+T}\cdot N' \\
&= \lambda_{1'}\lambda_{2'}(1-\lambda_{2'})^{T+1} \sum_{N'\geq 0} (1-\lambda_{2'})^{N'-1}\cdot N' \\
&= \lambda_{1'}\lambda_{2'}(1-\lambda_{2'})^{T+1}\left(\frac{1}{\lambda_{2'}}\right)^2 \\
&= \frac{\lambda_{1'}}{\lambda_{2'}}(1-\lambda_{2'})^{T+1}
\end{aligned}$$

□



(a) $B = \lambda_1 + \lambda_2 + \mu, N = 30$

(b) $B = 2(\lambda_1 + \lambda_2 + \mu), N = 60$

(c) $B = 5(\lambda_1 + \lambda_2 + \mu), N = 150$

(d) $B = 10(\lambda_1 + \lambda_2 + \mu), N = 300$

Fig. C.8. Various uniformization parameter B. WSN assignment (upper left, dark color) and DB assignment.

Appendix C. Optimal policy under different values of the uniformization parameter

The structure of the optimal policy for various values of the uniformization parameter B remains the same (see Fig. C.8). The validity threshold is set to $T = 1$.

References

[1] Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. Comput Netw 2002;38(4):393–422.
[2] Chen D, Varshney P. QoS support in wireless sensor networks: a survey. In: International Conference on Wireless Networks. 2004. p. 227–33.
[3] Lei J, Yates R, Greenstein L. A generic model for optimizing single-hop transmission policy of replenishable sensors. IEEE Transactions on Wireless Communications 2009;8(2):547–51.
[4] Munir A, Gordon-Ross A. An MDP-based application oriented optimal policy for wireless sensor networks. In: Proceedings of the 7th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis. 2009. p. 183–92.
[5] Krishnamachari B. Networking Wireless Sensors. Cambridge University Press; 2005.
[6] He T, Stankovic J, Lu C, Abdelzaher T. SPEED: a stateless protocol for real-time communication in sensor networks. In: Proceedings of the 23rd IEEE International Conference on Distributed Computing Systems. 2003. p. 6–55.
[7] Deb B, Bhatnagar S, Nath B. ReInForM: reliable information forwarding using multiple paths in sensor networks. In: Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks. 2003. p. 406–15.
[8] Stann F, Heidemann J. RMST: reliable data transport in sensor networks. In: Proceedings of the First International Workshop on Sensor Network Protocols and Applications. 2003. p. 102–12.
[9] Felemban E, Lee C, Ekici E, Boder R, Vural S. Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks. In: Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM. 2005. p. 2646–57.
[10] Yao Y, Gehrke J. Query processing in sensor networks. In: Proceedings of the 1st Biennial Conference on Innovative Data Systems Research. 2003.
[11] Khoury R, Dawborn T, Gafurov B, Pink G, Tse E, Tse Q, AlmiAni K, Gaber M, Röhm U, Scholz B. Corona: energy-efficient multi-query processing in wireless sensor networks. In: Proceedings of the Database Systems for Advanced Applications Conference. 2010. p. 416–9.
[12] Litjens R, Roijers F, van de Berg H, Boucherie RJ, Fleuren M. Performance analysis of wireless LANs: an integrated packet/flow level approach. In: Proceedings of the 18th International Teletraffic Congress. 2003.
[13] Hordijk A, van der Duyn Schouten FA. Discretization and weak convergence in Markov decision drift processes. Math Oper Res 1984;9(1):112–41.
[14] Dynkin EB. Markov Processes, vol. 1. Academic Press; 1965.
[15] Jensen A. Markoff chains as an aid in the study of Markoff processes. Scand Actuar J 1953;1953(Suppl. 1):87–91.
[16] Gikhman A, Skorokhod I. The Theory of Stochastic Processes: II, vol. 232. Springer; 2004.
[17] Van Dijk N. On a simple proof of uniformization for continuous and discrete-state continuous-time Markov chains. Adv Appl Probab 1990;22(3):749–50.
[18] van Dijk N, Hordijk A. Time-discretization for controlled Markov processes. I. General approximation results. Kybernetika 1996;32(1):1–16.
[19] Hordijk A, Schassberger R. Weak convergence for generalized semi-Markov processes. Stoch Proc Appl 1982;12(3):271–91.
[20] Puterman ML. Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons Inc.; 1994.
[21] Odoni AR. On finding the maximal gain for Markov Decision Processes. Oper Res 1969;17(5):857–60.