Computer Communications 109 (2017) 146–161

Contents lists available at ScienceDirect

Computer Communications

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

A generalized quorum system for asynchronous sleep scheduling in multi-hop wireless networks

Amir Hossein Saligheh[a], Vesal Hakami[b], Mehdi Dehghan[b,1,*]

[a] Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran.
[b] Department of Computer Engineering, Iran University of Science and Technology, PO Box 16846-13114, Hengam St., Resalat Sq., Tehran, Iran.

ARTICLE INFO

Article history:
Received 24 April 2016
Revised 11 February 2017
Accepted 7 June 2017
Available online 10 June 2017

Keywords:
Asynchronous sleep scheduling
Multi-hop networks
Quorum
Systems
Wireless ad-hoc networks
Wireless sensor networks

ABSTRACT

It has been shown that quorum systems with rotation closure property can be utilized to enable asynchronous power saving in wireless ad-hoc networks. Moreover, the quorum system characteristics (e.g., cycle length, activity ratio, overlap size) can be exploited to provide for a variety of sleep-scheduling paradigms which differ in terms of their adaptability and symmetry properties. To enable these paradigms, generally requires that the quorum schedules available to the nodes be heterogeneous so that they may pick their schedules in accordance with their roles or instantaneous conditions. Motivated by this heterogeneity requirement, in this paper, we generalize the basic notion of a quorum system into what we refer to as a Multi-Class Quorum System (MCQS). MCQS consists of a set of quorum classes, each of which is basically a quorum system with a guaranteed intra-class overlap size. Also, every two quorum classes have a guaranteed inter-class overlap size. We extend the classical rotation closure property of the quorum systems to guarantee any desirable number of intra- and inter-class overlaps between asynchronous nodes. We show that MCQS gives rise to a unified framework for quorum-based scheduling which can be easily tailored to enable all types of sleep-scheduling paradigms. We propose a binary integer programming (BIP) model to compute instances of MCQS which are optimal in terms of activity ratio. Also, unlike comparable quorum systems in the literature, MCQS is relieved of many restrictions on cycle length and overlap size. As a proof of concept, we study the application of MCQS in the joint problem of Quorum-based Sleep-scheduling and Routing (QSR) in wireless sensor networks (WSNs). We propose a centralized energy-optimal formulation as well as a decentralized algorithm for the QSR problem, and compare their performance through simulations.

1. Introduction

1.1. Research background

The recent technological advancements in wireless communications have extended the applications of wireless networks well beyond their original technological context and have led to many interesting communications scenarios. Among these, multi-hop wireless networks, especially with their low cost of deployment, have gained significant popularity in recent years. These networks run in an infrastructure-less fashion and are comprised of wireless nodes connected together through multi-hop paths. However, given that the constituent nodes subsist on limited battery supplies, energy scarcity has traditionally been the major impediment to the smooth operation of these networks. A well-established technique for prolonging the nodes' lifetime in multi-hop wireless networks is to deploy sleep-scheduling schemes for effectively minimizing the idle listening time in the nodes' network interface [1–5]. The existing sleep-scheduling schemes can be categorized into three groups [6]: on-demand wake-up, scheduled wake-up, and asynchronous wake-up.

In on-demand wake-up schemes [7,8], nodes wake up only when they have data to exchange. Two network interfaces are needed to serve this purpose: one for exchanging signaling messages, and the other for actual data communication. A node's signaling interface is always active, while its data interface gets activated only when another node calls upon through the signaling interface. The on-demand mechanism is still energy-efficient since this extra signaling interface operates with low power. However, its implementation can prove both costly and complicated. In scheduled wake-up schemes, all nodes wake up concurrently to perform

* Corresponding author at: Department of Computer Engineering and Information Technology, Amirkabir University of Technology, PO Box 15875-4413, 424 Hafez Avenue, Tehran, Iran.
E-mail addresses: amirhosseinsaligheh@aut.ac.ir (A.H. Saligheh), vhakami@iust.ac.ir (V. Hakami), dehghan@aut.ac.ir (M. Dehghan).
[1] URL: http://ceit.aut.ac.ir/~dehghan

http://dx.doi.org/10.1016/j.comcom.2017.06.002
0140-3664/

data exchange. In other terms, it is assumed that the nodes are synchronized in time. The well-known IEEE 802.11 PS [9] is an instance of scheduled wake-up schemes which is primarily designed for one-hop networks. The asynchronous wake-up schemes [10–12], on the other hand, relax the node synchronization assumption underlying the scheduled wake-up mechanisms. In these schemes, each node wakes up in pre-determined time intervals to communicate with its neighbors, and it is guaranteed that each node's wake-up pattern overlaps with those of its neighbors. These methods are particularly suited for cases where node synchronization is impossible or otherwise obtainable at great cost or with large overhead. Asynchronous schemes come with several benefits such as: ease of implementation, high scalability, and low overhead.

Among the asynchronous wake-up schemes, quorum-based algorithms have been shown to be the most effective [13], and thus have attracted a significant portion of the research interest. A quorum system is a group of non-empty subsets of a universal set, each called a quorum, whose pairwise intersection is non-empty. In quorum-based sleep-scheduling, the time axis is divided into fixed intervals, each called a *slot*. Every n consecutive slots constitute a *frame* and the universal set consists of all slot numbers within a frame. Each node picks an arbitrary quorum from the quorum system as its wake-up pattern. The elegance of the quorum-based scheduling lies in that the quorum system needs to satisfy a so-called *rotation closure property* so that all pairs of nodes are guaranteed to rendezvous despite being asynchronous at the slot level. The main characteristics of a quorum system which impacts sleep-scheduling algorithms are: *cycle length, activity ratio*, and *guaranteed overlap size*. Cycle length is determined by the cardinality of the universal set, and will amount to the frame length in TDMA parlance. The activity ratio of a given quorum is the ratio of the quorum size to the system's cycle length, and can be deemed as a measure of energy consumption by the node adopting that quorum as its sleep-awake schedule. The guaranteed overlap size refers to the minimum number of common elements between any two quorums of the given system, and will amount to the number of guaranteed overlapping awake slots between any two nodes adopting those quorums.

1.2. Motivation

Although quorum-based scheduling enables power saving in the context of asynchronous networks, requiring all nodes to adopt schedules with similar quorum-based parameters is problematic as individual nodes typically differ in terms of their topological conditions, and may have their specific demands on communications (e.g., delay, throughput, etc.) and constraints on resource availability (energy, bandwidth, etc.). This observation has resulted in a number of recent proposals [6,13–16] that aim at designing heterogeneous sleep scheduling algorithms to effectively manage the activity of the nodes in accordance with their specific roles and conditions. In fact, the quorum system characteristics can be exploited to derive a variety of asynchronous sleep-scheduling algorithms. These algorithms can be typified based on their (non)adaptability and (a) symmetry properties: in adaptive algorithms, each node, depending on its current conditions, manages to dynamically switch between quorum schedules with different activity ratios and guaranteed overlap sizes. In asymmetric schemes, the adopted quorums may be such that some pairs of nodes would not rendezvous at all. The non-adaptive and/or symmetric types can be characterized accordingly.

Despite the recognition of the need for heterogeneous quorum schedules, all existing schemes still consist of ad-hoc and problem-specific solutions. In fact, the existing quorum systems lack generality as each system is applicable only in one of the adaptive/non-adaptive and symmetric/asymmetric cases. Beyond that, all existing schemes are bound to restrictive assumptions because they impose an explicit constraint on quorum parameters (e.g., system size or guaranteed overlap size). Some only work with special system sizes (e.g., complete square as in [10,13,16,17,24]). In other cases (e.g., [12,13,18–21]), one can only obtain a fixed and limited guaranteed overlap size. There have been some efforts (e.g., HQS in [22]) to create quorum systems with varying cycle lengths (and therefore overlap sizes); but again, HQS does not support arbitrary overlap sizes for a particular system size.

1.3. Overview and contributions

Motivated by the heterogeneity requirement in quorum-based scheduling algorithms, and the lack of generality in existing schemes, in this article, we introduce a new quorum-based notion, namely *multi-class quorum system* (*MCQS*), which can overcome the shortcomings associated with the previous schemes. Our contributions can be summarized as follows:

- MCQS is a generalization of the conventional quorum system which can be exploited in all the afore-mentioned types of quorum-based sleep-scheduling algorithms. MCQS is comprised of a number of quorum classes. Each quorum class is essentially a quorum system with a specific number of intra-class overlap size (the guaranteed overlap size between any two quorums belonging to the class). Also, each pair of quorum classes have a specific number of inter-class overlap size (the guaranteed overlap size between any two quorums each belonging to one of the two classes). We extend the classical rotation closure property of the quorum systems in order to guarantee the desired intra- and inter-class overlap sizes between the asynchronous nodes. In addition, in MCQS, the quorums belonging to a given class are homogenous (i.e., they have the same quorum system characteristics), but the quorums belonging to different classes may be heterogeneous. Overall, MCQS is characterized by the following parameters: cycle length, activity ratio, number of classes, intra-class overlap size, and inter-class overlap size. We will show that with the tuning of its parameters, MCQS is amenable to customized use in all types of sleep-scheduling algorithms.
- We realize that existing quorum systems mostly restrict both their cycle length (e.g., square or prime number) and their guaranteed overlap size. Also, some of these systems are non-optimal in terms of their activity ratio. Using binary integer programming (BIP), we propose one way to compute MCQS which ensures optimality in terms of activity ratio. The resultant MCQS is also relieved of above restrictions.
- As a proof of concept for our generalized quorum construct, we implement an MCQS-based asynchronous sleep-scheduling algorithm for a wireless sensor network (WSN). In this algorithm, the nodes may choose their quorum schedule from either one of the available classes depending on their conditions. In WSNs, the traffic load experienced by the nodes is a major determining factor for quorum class selection. The traffic load on each node depends largely on the paths prescribed by the underlying routing scheme. On the other hand, the quorum class adopted by the nodes can mutually affect the routing performance. Armed with this understanding, in this paper, we first leverage on both MCQS and cross-layering to present a BIP model for the joint problem of *Q*uorum-based *S*leep-scheduling and *R*outing (QSR) in WSNs. The solution is optimal in terms of energy consumption, while also satisfying the data rate demands mandated by the routing. We also propose a distributed algorithm (DQSR) for the QSR problem. In DQSR, to decentralize the quorum class selection process, a so-called overlap graph is first constructed on top of the topological network. We

then compute the routes jointly with the nodes' schedules by applying a distributed minimum-weight vertex-disjoint paths algorithm over the constructed overlap graph. As evidenced by the simulation experiments, DQSR is capable of achieving near-optimal performance for the QSR problem in WSNs.

1.4. Outline

The rest of the paper is organized as follows: in Section 2, we survey the state of the art in quorum-based asynchronous sleep scheduling, and give a taxonomy of the existing algorithms. In 3, we introduce MCQS, our generalized quorum-based notion, and will present a BIP model for its computation. As a proof of concept for MCQS, in Section 4, we present a BIP formulation of the QSR problem. We then discuss DQSR, our distributed and computationally efficient approach for solving QSR. In 5, we give numerical results for the MCQS performance, and will also evaluate our solutions for the QSR problem. The paper concludes in Section 6.

2. Related work

In one broad taxonomy, quorum-based sleep scheduling algorithms can be categorized as either *homogenous* or *heterogeneous*. In homogeneous sleep-scheduling algorithms, the quorums adopted by all nodes are identical in terms of their characteristics (e.g., cycle length, activity ratio, and guaranteed overlap size). On the other hand, in heterogeneous sleep-scheduling algorithms, the nodes' sleep schedule is determined by quorums with non-identical characteristics. In one particular case, some nodes may not rendezvous at all given that their adopted quorum schedule may be non-intersecting. This particular subcategory of heterogeneous sleep-scheduling algorithms is referred to as *asymmetric* algorithms. *Adaptive* algorithms constitute another subcategory of heterogeneous scheduling schemes. In these algorithms, the nodes can dynamically switch between a number of heterogeneous quorum schedules depending on their operational conditions.

By virtue of this taxonomy, QPS schemes fall into either one of the following four categories: *non-adaptive symmetric, non-adaptive asymmetric, adaptive symmetric* and *adaptive asymmetric*. Below, we give a succinct overview of the most representative schemes from each category, and point out their shortcomings:

- **Non-adaptive symmetric:** The Quorum-based Protocol [10] is the pioneering work in the context of asynchronous quorum-based wake-up scheduling schemes. The analytical guarantee given by this protocol is the existence of at least two overlapping time slots in between every pair of nodes, which is established at the expense of a $(2\sqrt{n}-1)/n$ activity ratio. Quorum-based protocol is built on the notion of a Grid quorum system [13,18] in which each node arbitrarily chooses one row and one column from a $\sqrt{n}\times\sqrt{n}$ square grid as its wake-up pattern. Much in the same way as all Grid-based schemes, the quorum system size in [10] is limited to perfect squares. A further drawback is that the activity ratio in this protocol is nearly twice the theoretical optimum. The Cyclic Difference Set-based Protocol (CDS) protocol in [12,19] is another instance of non-adaptive symmetric quorum-based schemes. Compared to [10], the sleep-awake schedule induced by CDS is more efficient in terms of energy consumption. CDS works on the basis of cyclic quorum systems which are in turn founded on the notion of difference sets in combinatorial mathematics. The activity ratio in CDS coincides with the theoretical optimum; moreover, every two nodes are guaranteed to have at least one rendezvous during a given cycle. However, as was the case in [10], the cycle length in CDS cannot be chosen freely.
- **Non-adaptive asymmetric:** The Asymmetric Cyclic Quorum (ACQ) protocol [20,21] has been the first asymmetric quorum-based wake-up scheduling scheme. ACQ is specialized for use in a clustered ad-hoc network and provisions for two energy saving regimes across the nodes. More precisely, the quorum system used in [20,21] consists of two types of quorums: the *s*-quorums to be adopted by cluster heads and *a*-quorums which are intended for cluster members. The analytical guarantee given by this asymmetric paradigm is that nodes withs-quorum schedule will rendezvous both with other *s*-quorum nodes as well as with those having an *a*-quorum-based schedule. This is while no rendezvous is guaranteed between two nodes adopting *a*-quorum schedules. A drawback associated with ACQ is that it does not prescribe how to efficiently construct asymmetric quorums with arbitrary cycle lengths.
- **Adaptive symmetric:** The major flaw with non-adaptive schemes is that the nodes would not be able to adjust the activity ratio in their sleep-awake schedule in accord with their traffic load, position in topology and mobility conditions. In order to introduce adaptability to quorum-based scheduling, the Adaptive Quorum-based Power-Saving (AQPS) [13] and Adaptive Quorum-based Energy Conserving (AQEC) [17] protocols allow for the nodes to fine-tune their energy consumption level by individually selecting their quorum cycle lengths. The rendezvous guarantee is still maintained, but the cycle length is limited to perfect squares and the activity ratio is nearly twice the theoretical optimum. The Adaptive Asynchronous Power Management (AAPM) protocol [15] extends the CDS idea to feature adaptability. Basically it constructs a pool of ordered pairs, namely AA-quorum space, which consists of quorums with different cycle lengths. Every two quorums within the AA-quorum space are guaranteed to intersect. Despite achieving an optimal activity ratio, the eligible cycle lengths for the AA-quorum space are restricted to prime numbers. Also, AAPM does not scale efficiently for large cycle lengths of $n > 47$. Hyper Quorum System (HQS) [22] is basically a generalization of traditional quorum systems providing for quorums with arbitrary cycle lengths which can be flexibly used by nodes in the network. HQS draws on a projection concept and maps quorums with varying cycle lengths into a modulo-*m*-plane. It is then guaranteed that these projected quorums have at least one overlap with each other. Extended Grid HQS (EGHQS) and Difference Set HQS (DSHQS) are two algorithms which can compute HQS. However, they do not support arbitrary overlap sizes for a particular system size. Despite their adaptability, the activity ratio in EGHQS is almost as high as that in AQEC [17] and DSHQS's activity ratio is near-optimal only for small cycle lengths of $n \leq 25$. Also the projection algorithm in HQS grows inefficient as n increases. Recently, the work in [22] has been extended in [23] to describe how to find the optimal cycle length for a node to maximize energy efficiency, subject to certain performance constraints**.** Finally, the Adaptive Cyclic Difference Set (ACDS) protocol [6] is a recent proposal in the area of adaptive QPS schemes. In ACDS, a hierarchy of difference sets are used to enable varying degrees of power saving. However, ACDS is primarily proposed for large cycle length systems, and in particular for delay tolerant networks (DTNs).
- **Adaptive asymmetric:** The Asynchronous, Adaptive, and Asymmetric (AAA) power management protocol [24] is essentially a synergistic blend of the ideas underlying HQS [22] and ACQ [20,21] protocols. Much in the same way as HQS, each node using AAA can adopt a Grid-based quorum with a cycle length most apt with its current conditions. Moreover, in a similar way to ACQ, AAA provides for asymmetry through *s*-quorum and *a*-quorum schedules. The *s*-quorum is comprised of one row and one column chosen arbitrarily from a $\sqrt{n}\times\sqrt{n}$ square grid. The *a*-quorum, on the other hand, only consists of the slot numbers within a single column. This way, each node is free to choose either an *a*-quorum or an *s*-quorum schedule with

a cycle length well-suited for its condition. However, given the AAA's Grid-based design, the nodes would not benefit from an optimal activity ratio. Also, the choice for cycle length is again restricted to perfect squares. The Dygrid system in [16] can also be used to derive adaptive and asymmetric sleep-scheduling algorithms. Dygrid is comprised of two types of quorums: h-cliques and v-cliques. In Dygrid, it is guaranteed that each h-clique quorum intersect with a v-clique quorum. This is while h-cliques or v-cliques themselves may not intersect. While Dygrid is optimal in terms of activity ratio, its cycle length, however, is bound to perfect squares.

3. Multi-class quorum system (MCQS)

In this section, we build on the classical definition of a quorum system to introduce a new generalized notion, namely, multi-class quorum system (MCQS) along with its extended rotation closure property. MCQS extends the definition of the quorum systems and can be customized for application in all types of asynchronous quorum-based sleep-scheduling algorithms. We then propose a BIP model in 3.2 to compute an MCQS which is optimal in terms of quorum activity ratio. Also, MCQS has no restriction with respect to the cycle length and the guaranteed overlap size between all pairs of quorums. Finally, in 3.3, we show how MCQS can be specialized to support all varieties of sleep scheduling algorithms.

3.1. Definitions

Before introducing MCQS, we first define the notion of a quorum class.

Definition 1. For the universal set $U = \{0, 1, \ldots, n-1\}$, a set QC of quorums in U is called a quorum class with cycle length n and guaranteed overlap size C if the following two conditions hold:

$$\forall Q, Q' \in QC \ : \ \left|Q \cap Q'\right| \geq C$$

$$\forall Q, Q' \in QC \ : \ AR(Q) = AR\left(Q'\right)$$

The symbol C is a positive integer and denotes the minimum number of common elements between any two quorums in QC. $AR(Q)$ is the activity ratio of the quorum Q and is defined as the ratio of the cardinality of Q to the cycle length n. Since by definition, all the quorums belonging to a given class have the same activity ratio, we may consider AR as a class characteristic. Finally, it is worth mentioning that by construction, a quorum class is also a quorum system.

Definition 2. For the universal set $U = \{0, 1, \ldots, n-1\}$, a set $QS^m = \{QC_1, QC_2, \ldots, QC_m\}$ is called an m-class quorum system under U if QC_1, QC_2, …, QC_m are quorum classes with cycle length n and with guaranteed overlap sizes C_1, C_2, …, C_m and the following condition holds:

$$\forall i, j \in [1, \ldots, m],\ Q \in QC_i,\ Q' \in QC_j \ : \left|Q \cap Q'\right| \geq C_{i,j}$$

In the above definition, QC_i is called the i-th class of the MCQS QS^m. $C_{i,j}$ is a positive integer and denotes the minimum number of common elements between any quorum Q of the i-th class and a quorum Q' of the j-th class. We henceforth refer to $C_{i,j}$ as the inter-class overlap size between the classes QC_i and QC_j. Accordingly, for each quorum class QC_i, C_i denotes its intra-class overlap size. Based on this definition, in MCQS, all the quorums of a given class are homogenous, but the quorums belonging to different classes may be heterogeneous and have different characteristics.

These definitions will be made clearer by an example: consider the universal set $U = \{0, 1, \ldots, 6\}$. $QC_1 = \{\{2, 3\}, \{2, 4\}\}$ is a quorum class with cycle length 7 containing two quorums {2, 3} and {2, 4}. Also, $QC_2 = \{\{0, 2, 3, 6\}, \{0, 1, 2, 5\}, \{0, 2, 3, 4\}\}$ is a quorum class with three quorums {0, 2, 3, 6}, {0, 1, 2, 5} and {0, 2, 3, 4}. $QS^2 = \{QC_1, QC_2\}$ is a 2-class quorum system under U. The intra-class overlap size for the class QC_1 is $C_1 = 1$, and for the class QC_2 is $C_2 = 2$. Also, the inter-class overlap size for the pair QC_1 and QC_2 is $C_{1,2} = 1$. The activity ratios of the classes QC_1 and QC_2 is $\frac{2}{7}$ and $\frac{4}{7}$, respectively.

Definition 3. Similarly to the conventional quorum system [13], the r-th rotation of a quorum Q of any class is defined as follows:

$$Rotate(Q, r) = \{(t + r) \ mod \ n \mid t \in Q\}$$

Obviously, $Rotate(Q, 0) = Q$.

Definition 4. The MCQS QS^m under $U = \{0, 1, \ldots, n-1\}$ has the rotation closure property if:

$$\forall i, j \in [1, \ldots, m],\ Q \in QC_i,\ Q' \in QC_j,$$
$$r \in [0, \ldots, n-1] \ : \ Q \cap Rotate\left(Q', r\right) \neq \emptyset$$

Now, in order to guarantee C_i common elements between any two quorums from the i-th class, $i = 1, \ldots, m$ and to also guarantee $C_{i,j}$ common elements between an i-th class quorum and a j-th class quorum, $i, j = 1, \ldots, m,\ i \neq j$, we need to extend the rotation closure property for the MCQS QS^m:

Definition 5. MCQS QS^m under $U = \{0, 1, \ldots, n-1\}$ has the extended rotation closure property if:

$$\forall i \in [1, \ldots, m],\ Q, Q' \in QC_i,$$
$$r \in [0, \ldots, n-1] : \ \left|Q \cap Rotate\left(Q', r\right)\right| \geq C_i$$

$$\forall i, j \in [1, \ldots, m], j \neq i,\ Q \in QC_i,\ Q' \in QC_j,$$
$$r \in [0, \ldots, n-1] : \ \left|Q \cap Rotate\left(Q', r\right)\right| \geq C_{i,j}$$

3.2. Computing an optimal MCQS

In this section, we present a BIP formulation for computing an MCQS construct with extended rotation closure property. The computed MCQS has cycle length n, contains m classes with intra-class overlap size $C_i,\ i = 1, 2, \ldots, m$ and inter-class overlap size $C_{i,j},\ i, j = 1, 2, \ldots, m$ and $j \neq i$. Also, each class QC_i has h_i quorums. These parameters can be given as input to the formulation in 3.2.2. The computed MCQS is optimal in terms of quorum activity ratio.

3.2.1. Notations

Let $QS^m = \{QC_1, QC_2, \ldots, QC_m\}$ be an MCQS under $U = \{0, 1, \ldots, n-1\}$ with cycle length n. Also, assume that $QC_i = \{Q_i^1, Q_i^2, \ldots, Q_i^{h_i}\}$ is the i-th class of QS^m containing h_i quorums. Q_i^k is the k-th quorum of the i-th class.

The decision variables used in our formulation are as follows:

- The inclusion of $t \in U$ in quorum Q_i^k is indicated by $W_i^k(t)$:

$$W_i^k(t) = \begin{cases} 1, & t \in Q_i^k \\ 0, & t \notin Q_i^k \end{cases}$$

- The inclusion of $t \in U$ in the r-th rotation of quorum Q_i^k is indicated by $R_i^k(t, r)$:

$$R_i^k(t, r) = \begin{cases} 1, & t \in Rotate\left(Q_i^k, r\right) \\ 0, & t \notin Rotate\left(Q_i^k, r\right) \end{cases}$$

Obviously, $R_i^k(t, 0) = W_i^k(t)$.

- Also, we introduce the binary variable $I_{i,j}^{k,l}(t, r)$ to indicate the inclusion of $t \in U$ in the intersection of Q_i^k with the r-th

Objective Function:

$$Minimize \ \frac{1}{n}\sum_{i}\sum_{k}\sum_{t} W_i^k(t) \quad (1)$$

Subject to:

$$R_i^k(t,r) = W_i^k((t-r) mod\ n) \quad \forall\, i \in [1,\dots,m], k \in [1,\dots,h_i], r \in [0,\dots,n-1], t \in [0,\dots,n-1] \quad (2)$$

$$W_i^k(t) + R_j^l(t,r) - 2I_{i,j}^{k,l}(t,r) \geq 0 \quad \forall\, i,j \in [1,\dots,m], k \in [1,\dots,h_i], l \in [1,\dots,h_j], r \in [0,\dots,n-1], t \in [0,\dots,n-1] \quad (3)$$

$$W_i^k(t) + R_j^l(t,r) - 2I_{i,j}^{k,l}(t,r) \leq 1 \quad \forall\, i,j \in [1,\dots,m], k \in [1,\dots,h_i], l \in [1,\dots,h_j], r \in [0,\dots,n-1], t \in [0,\dots,n-1] \quad (4)$$

$$\sum_{t} I_{i,i}^{k,l}(t,r) \geq C_i \quad \forall\, i \in [1,\dots,m], k,l \in [1,\dots,h_i], k \neq l, r \in [0,\dots,n-1] \quad (5)$$

$$\sum_{t} I_{i,j}^{k,l}(t,r) \geq C_{i,j} \quad \forall\, i,j \in [1,\dots,m], i \neq j, k \in [1,\dots,h_i], l \in [1,\dots,h_j], r \in [0,\dots,n-1] \quad (6)$$

$$\frac{1}{n}\sum_{t} W_i^k(t) = \frac{1}{n}\sum_{t} W_i^l(t) \quad \forall\, i \in [1,\dots,m], k,l \in [1,\dots,h_i] \quad (7)$$

Fig. 1. BIP model for computing an MCQS with extended rotation closure property.

rotation of Q_j^l; i.e., $I_{i,j}^{k,l}(t,r) = 1$ if $t \in Q_i^k \cap Rotate(Q_j^l, r)$. in other words,

$$I_{i,j}^{k,l}(t,r) = \begin{cases} 1, & W_i^k(t) = 1 \text{ and } R_j^l(t,r) = 1 \\ 0, & W_i^k(t) = 0 \text{ or } R_j^l(t,r) = 0 \end{cases}$$

In case $j = i$, $I_{i,j}^{k,l}(t,r)$ indicates the inclusion of $t \in U$ in the intersection of Q_i^k with the r-th rotation of its classmate Q_i^l. In particular, for $r = 0$, $I_{i,j}^{k,l}(t,0)$ indicates the inclusion of $t \in U$ in the intersection of Q_i^k with Q_j^l.

3.2.2. A BIP formulation for computing MCQS

The proposed BIP model for computing an MCQS with extended rotation closure property is given in Fig. 1. By solving this model, the quorums belonging to each class can be determined; in fact, each quorum Q_i^k consists of a subset of elements $t \in U$ for which $W_i^k(t) = 1$. To ensure the optimality of the resulting MCQS in terms of activity ratio, we define the objective function as the sum of the quorum's activity ratios (see (1)).

The constraint (2)defines the variable $R_i^k(t,r)$ in terms of the variable $W_i^k(t)$. In other words, the constraint (2) indicates the inclusion/exclusion of $t \in U$ in the r-th rotation of quorum Q_i^k.

The constraints (3) and (4) define the variable $I_{i,j}^{k,l}(t,r)$ in terms of the variables $W_i^k(t)$ and $R_j^l(t,r)$; i.e., these two constraints indicate together whether $t \in U$ is a member of the set $Q_i^k \cap Rotate(Q_j^l, r)$.

The constraint (5) for each class QC_i is on the intra-class overlap size. For $r = 0$, This constraint ensures that at least C_i overlaps exist between any two arbitrary quorums from class QC_i. For $r \neq 0$, it ensures that at least C_i overlaps exist between any quorum from QC_i and all the rotations of its classmate quorums.

The constraint (6) is on the inter-class overlap size for each pair of classes QC_i and QC_j; more specifically, for $r = 0$, this constraint ensures that at least $C_{i,j}$ overlaps exist between any quorum from class QC_i and an arbitrary quorum from class QC_j; likewise, for $r \neq 0$, it is ensured that at least $C_{i,j}$ overlaps exist between any quorum from class QC_i and the r-th rotation of any quorum from class QC_j. It is worth mentioning that when $r \neq 0$, the two constraints (5) and (6) ensure the extended rotation closure property in QS_m.

The constraint (7) guarantees that any two quorums from the same class have identical activity ratios.

3.2.3. A note on computational cost

Computing MCQS is a generalization of the classical quorum system computation with arbitrary parameters, which has no precedence in the literature. To solve this problem in this paper, we have come up with a standard BIP formulation. We understand that solving a BIP has a worst-case exponential time complexity, but there might be room for proposing a heuristic algorithm to compute a near-optimal MCQS with reduced complexity. However, there are other reasons which mitigate the concerns for practicability of our BIP method:

- First, in sleep scheduling algorithms, quorum system sizes (which correspond to TDMA frame length) are normally small values (e.g., <= 100), and the BIP problem underlying MCQS calculation would be manageable.
- Second, in order to employ MCQS in a sleep scheduling algorithm, it suffices to compute an optimal MCQS only once and store it for subsequent reuse.

Overall, as "activity ratio" is the most critical factor in sleep scheduling protocols, it would be justifiable to pay a small price in terms of computational cost to gain significantly in terms of network lifetime.

3.3. MCQS-based sleep-scheduling

MCQS-based sleep-scheduling requires that each node select its quorum (class) schedule from the available pool of classes QC_1, QC_2, …, QC_m all belonging to a certain MCQS instance QS^m with extended rotation closure property. Consider two nodes A and B in an asynchronous wireless system. Assume that node A (resp. B) chooses quorum Q (resp. Q') as its schedule, where Q and Q' both belong to MCQS QS^m. Then, it is guaranteed that:

- A and B have C_i overlapping awake slots if Q and Q' belong to the same class QC_i in which case A and B would also operate with the same activity ratio.

or:

- A and B have $C_{i,j}$ overlapping awake slots if $Q \in QC_i$, $Q' \in QC_j$ and $i \neq j$ in which case A and B would operate with different activity ratios.

The notion of quorum classification embedded within MCQS makes it customizable for use in different types of sleep-scheduling algorithms. This customization is done through appropriate tuning of the MCQS parameters:

In case $m = 1$, MCQS reduces to the traditional single-class quorum system. This system can be utilized in non-adaptive symmetric sleep-scheduling algorithms. For instance, with $m = 1$ and $C_1 = 1$, MCQS mimics a cyclic quorum system [13,25]. Also, if $m = 1$, $C_1 = 2$, and n is a square number, MCQS is similar to a grid quorum system [13,18], albeit with an optimal activity ratio.

In order to customize MCQS for asymmetric or adaptive algorithms, we need at least $m \geq 2$ classes. These classes are non-homologous in terms of their intra- and inter-class overlap sizes.

Also, the quorums belonging to different classes have different activity ratios.

The wireless nodes scheduled with an adaptive algorithm can dynamically switch between different quorum classes depending on their current working conditions. For instance, the classes QC_1 and QC_2 of an MCQS with $m = 2$, $C_1 = 1$, $C_2 = 2$, and $C_{1,2} = 1$ are similar to e-torus(1) [13,26] and e-torus(2), respectively.

Finally, in asymmetric algorithms, some quorums need not intersect at all, which is somewhat contradictory to the original definition of quorum systems [13]. Examples are read/write quorums in [27] or the notions of s-quorum and a-quorum in ACQ [20,21]. In order to customize MCQS for these types of algorithms, it suffices to set the intra-class overlap size to zero. In other words, an MCQS with $m = 2$, $C_1 = 1$, $C_2 = 0$, and $C_{1,2} = 1$ is equivalent to read/write quorums, in which QC_1 includes the write quorums and QC_2 includes the read quorums.

4. MCQS-based sleep scheduling in wireless sensor networks

A wireless network typically consists of non-homogeneous nodes; in fact, wireless nodes differ in terms of their topological conditions, energy availability, traffic load, etc. This non-homogeneity calls for heterogeneous sleep scheduling algorithms to effectively manage the activity of the nodes in accordance with their specific conditions. In this section, as a proof-of-concept for our proposed MCQS notion, we demonstrate how it can be utilized to enable heterogeneous scheduling for wireless nodes. In fact, MCQS already features a collection of quorum classes with different activity specifications. This feature can be readily exploited to translate the heterogeneous scheduling problem into one of quorum class selection. Quorum class selection can be performed based on different criteria: for instance, the nodes may choose their schedule depending on their operational role in the network. Consider a clustered wireless network as a typical example. The cluster heads should adopt their quorum from a class with a higher activity ratio and a larger overlap size; this is while the cluster members, given their less critical role in the network, may be scheduled using a class with less demanding activity and overlap specifications. Other criteria underlying quorum class selection could be the nodes' characteristics such as velocity, energy, load, etc. For instance, in a vehicular network, high velocity nodes should choose their quorum-based schedule from a class with a higher activity ratio and a larger overlap size to keep up with the rapid changes in their neighbor sets.

In wireless sensor networks, an important factor for determining the nodes' sleep/awake schedule would be the traffic load on the nodes. The specifics of the traffic load heavily depend on the underlying routing algorithm; in particular, the nodes lying on high traffic routes are required to adopt a more active class; conversely, less busy routes can live with a less demanding quorum class schedule. In turn, the scheduling algorithm can mutually affect the routing's performance; for instance, in case the nodes on a given route choose their quorum schedule from classes with higher activity ratio and larger overlap size, the data flow on that route incurs smaller latency. Hence, an efficient solution should take into account the interplay between routing and quorum class selection.

In 4.2, we take on a cross-layer approach and formulate the joint problem of quorum-based sleep-scheduling and routing (QSR) in wireless sensor networks as a BIP model. Our solution to the QSR problem heavily draws on the MCQS notion introduced in Section 3. In QSR, the goal is to determine both the routes and the nodes' quorum (class) schedules such that the sum of the nodes' activity ratios is minimized, while the data rate requirements of all the source nodes are satisfied. To accomplish this, the quorum adopted by each node should guarantee the on-time relaying of the packets passing through it. In Section 4.3, we also come up with a distributed algorithm for the QSR problem which, as evidenced by the simulation results in Section 5, is capable of achieving near optimal performance.

4.1. Assumptions

Consider a wireless sensor network consisting of a set $V = \{v_1, v_2, \ldots, Sink\}$ of sensors along with a sink node. The sensor nodes are typically tasked with environmental sensing and reporting the sensed data to the sink. Since in many WSN applications the data flows are associated with long-term sessions, it is often possible to estimate the average rate from each source node to the sink [28]. In particular, we assume that the source nodes sample from the environment periodically and send their data in the form of packets. Different source nodes operate with different sensing periods. Let T be the largest sensing period across all the source nodes. We use T_s to denote the sensing period of the source node v_s. Then, v_s sends $R_s = T/T_s$ packets in every T time interval. We call R_s the transmission rate of the source node v_s. We also assume that the time is divided into equal slots. Each n consecutive slots are called a frame. We consider the length of each frame to be equal to the largest transmission period T of the source nodes. Let t_s denote the length of each slot. We may calculate the cycle length as $n = T/t_s$. It is further assumed that each node can send only one packet in each time slot.

Each node is equipped with a half-duplex transceiver and an omni-directional antenna with a limited transmission range. The nodes located within the transmission range of node v_i are called its neighbors. Only a given node's neighbors can receive its transmissions. It can be assumed that each sensor node v_i is connected to its neighbor v_j via a link $\langle v_i, v_j \rangle$. We denote by $E = \{\langle v_i, v_j \rangle\}$ the set of all links in the network. Hence, the topology of a WSN can effectively be represented by a graph $G = \langle V, E \rangle$. Given the nodes' limited transmission range, the source nodes have to relay their packets through their neighbors in a multi-hop fashion.

In the absence of synchronization, the nodes are assumed to operate on the basis of a quorum-based asynchronous sleep-scheduling algorithm with cycle length n. Given the nodes' non-homogeneity in terms of their traffic load, we seek to deploy a scheduling mechanism using a heterogeneous variant of MCQS. We assume that the utilized MCQS $QS^m = \{QC_1, QC_2, \ldots, QC_m\}$ consists of m different classes, and that the nodes may adopt their quorum schedules from either one of these classes. It is worth mentioning that since all the quorums within a given class have equivalent activity ratio and overlap size, it suffices to only determine the appropriate class for each node. With a slight abuse of notation, let Q_k denote an arbitrary quorum from the k-th class. In case two nodes adopt the quorums Q_k and Q_l, respectively, their awake slots will coincide at least $C_{k,l}$ times; likewise, if a representative quorum Q_k from the k-th class is adopted by both nodes, their awake slots will coincide at least C_k times.

Table 1 lists the symbols and definitions used in QSR problem.

4.2. A centralized solution to QSR problem

In this section, we formulate the QSR problem as a BIP model. We refer to the solution as centralized QSR (CQSR for short). CQSR is optimal in terms of the nodes' total activity ratio, and can be used as a benchmark to evaluate our distributed solution in Section 5.2.

For ease of presentation, we first enrich the topological graph of the network with the nodes' quorum schedule information. Using this extended graph, we can define appropriate decision variables for our BIP model.

Table 1
Notations used in QSR problem.

Symbol	Description
V	Set of all nodes (sensors and sink)
$v_i \in V$	i-th sensor node
E	Set of all communication links in WSN
$\langle v_i, v_j \rangle \in E$	Communication link between sensor nodes v_i and v_j
$G = \langle V, E \rangle$	Network topological graph
S	Set of all source nodes
$v_s \in S$	s-th source node
R_s	data rate of source node v_s (packet/frame)
T	Largest source to sink transmission period
QS^m	An m-class MCQS instance
n	Number of slots in a frame (Cycle Length)
m	Number of classes in MCQS QS^m
Q_k	A class-k quorum
$AR(Q_k)$	Q_k's activity ratio Quorum
C_k	Intra-class overlap size for class QC_k
$C_{k,l}$	Inter-class overlap size between classes QC_k and QC_l

4.2.1. Extended topological graph

In order to find the flow routes from source nodes to the sink conjointly with the quorum classes best fitting the nodes' traffic conditions, we annotate each node in graph G with the set of all available quorums (from classes: QC_1, QC_2, ..., QC_m). This way, the topological graph $G = \langle V, E \rangle$ is extended to a new logical graph $G' = \langle V', E' \rangle$ which, besides the topological information, contains the candidate set of quorum (class) schedules for each node. More specifically, we replicate each node in topological graph m times (i.e., the number of quorum classes in MCQS QS^m). We refer to these annotated nodes in G' as quorum vertices. Hence, for each node $v_i \in V$ in graph G, there exists a set of vertices $\{v_{i,k}\}_{k \in [1, ..., m]}$ in the extended graph G'; likewise, for each link $\langle v_i, v_j \rangle \in E$ in G, there exists the set of edges $\{\langle v_{i,k}, v_{j,l} \rangle\}_{k,l \in [1,...,m]}$ in G'; in other terms, we connect each quorum vertex of a given node to all the quorum vertices of its neighbors. In Fig. 2(a), we show the topological graph of a network composed of four sensor nodes. In Fig. 2(b), we illustrate the corresponding extended graph under the assumption that a 2-class quorum system is used.

To determine the flow routes jointly with the optimal quorum (class) schedules of the nodes, we consider a specific decision variable for each source-sink path and for each edge in the extended graph G'. More specifically, we associate a binary decision variable $L_{i,k}^{j,l}(s)$ with each $v_s \in S$, and every $\langle v_{i,k}, v_{j,l} \rangle \in E'$. $L_{i,k}^{j,l}(s)$ is 1 if the link $\langle v_i, v_j \rangle$ lies on the path from v_s to the sink, and v_i and v_j have adopted the quorums Q_k and Q_l, respectively. We also introduce a binary variable P_i^k to indicate whether or not quorum Q_k is adopted by node v_i.

4.2.2. BIP formulation

The proposed formulation for the QSR problem is listed in Fig. 3.

Since we aim to minimize the nodes' energy consumption, the objective function is defined to be the sum of the nodes' activity ratios (see (8)). The formulation contains two groups of constraints: one on routing and one on quorum class selection:

Routing constraints. In QSR, we have to meet the rate required by each source node v_s. More specifically, v_s should be able to send R_s packets to the sink within each frame. Therefore, we establish a route (not necessarily dedicated) for each of the R_s packets of v_s. To this end, the following two conditions should hold in the extended graph G':

(i) one of the outgoing edges from one of the quorum vertices of v_s is activated.
(ii) in each quorum vertex (except for those of the sink's and v_s's) if an incoming edge is active, one outgoing edge should also be activated. This condition ensures the connectivity of the path from v_s to the sink.

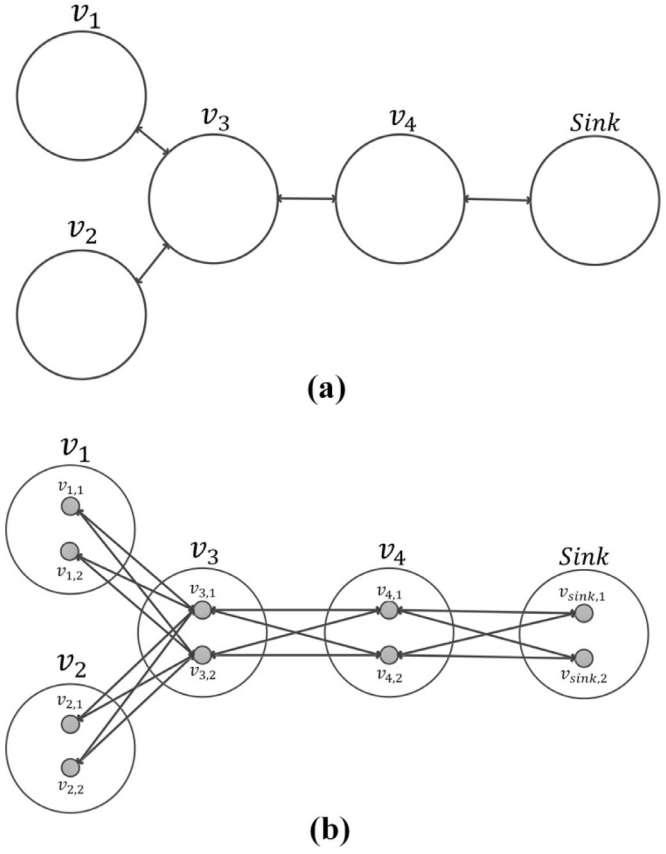

Fig. 2. Extended graph construction from the topological graph. (a) Topological graph G = ⟨V, E⟩. (b) Extended graph G' = ⟨V', E'⟩.

The constraints (9) and (10) handle the above two conditions for each of the routes between v_s and sink during a frame. However, it is still possible that a route from one of v_s's quorum vertices cycles back to another vertex of v_s. The constraint (11) is included to avoid this undesirability. Another loopy condition may occur between intermediate quorum vertices along a given a route. However, since each edge activation in G' would contribute to the required overlaps between the two associated quorum vertices, such intermediate loopy conditions are automatically pruned by enforcing the objective function.

Quorum class selection constraints. The constraints (12) and (13) determine the value of P_i^k with respect to the number of activated incoming and outgoing edges in the quorum vertices associated with node v_i. In (12) and (13), $\mathcal{L}$ is taken to be a large enough number[1] The constraint (14) ensures that at most one quorum will be selected by a node v_i. In case a node adopts no quorum at all, it will constantly remain inactive. The constraint (15) is to ensure that the number of overlapping awake slots between any two nodes scheduled with quorums from the same class does not exceed the intra-class overlap size; likewise, (16) is there to make sure that for any two nodes scheduled with quorums from the different classes, the number of overlapping awake slots is no more than the corresponding inter-class overlap size.

[1] it suffices to choose $\mathcal{L} \geq \Delta \times C^*$, where Δ denotes maximum node degree in graph G; i.e., $\Delta = \max_{v_i \in V} |\{v_j\}_{v_i, v_j \in E}|$. Also, C^* is defined to be the maximum number of possible overlaps across all quorum classes within the given MCQS; i.e., $C^* = Max\{C_k, C_{k,l}\}$, $\forall\, k, l \in \{1, 2, \ldots, m\}$.

Objective Function:

$$Minimize \sum_{i|v_i \in V} \sum_{k \in [1,\ldots,m]} A_k P_i^k \tag{8}$$

Subject to:

$$\sum_{j|\langle v_s,v_j\rangle \in E} \sum_{k \in [1,\ldots,m]} \sum_{l \in [1,\ldots,m]} L_{s,k}^{j,l}(s) = R_s \qquad \forall v_s \in S \tag{9}$$

$$\sum_{j|\langle v_i,v_j\rangle \in E} \sum_{l \in [1,\ldots,m]} L_{i,k}^{j,l}(s) - \sum_{j|\langle v_j,v_i\rangle \in E} \sum_{l \in [1,\ldots,m]} L_{j,l}^{i,k}(s) = 0 \qquad \forall v_s \in S,\ \forall v_i \in V - \{v_s, sink\},\ \forall Q_k \in QS^m \tag{10}$$

$$\sum_{j|\langle v_j,v_s\rangle \in E} \sum_{k|Q_k \in QS^m} \sum_{l|Q_l \in QS^m} L_{j,k}^{s,l}(s) = 0 \qquad \forall v_s \in S \tag{11}$$

$$P_i^k - \frac{1}{\mathcal{L}} \left(\sum_{s|v_s \in S} \sum_{j|\langle v_i,v_j\rangle \in E} \sum_{l|Q_l \in QS^m} L_{i,k}^{j,l}(s) + \sum_{s|v_s \in S} \sum_{j|\langle v_j,v_i\rangle \in E} \sum_{l|Q_l \in QS^m} L_{j,l}^{i,k}(s) \right) \geq 0 \qquad \forall v_i \in V,\ \forall Q_k \in QS^m \tag{12}$$

$$P_i^k - \frac{1}{\mathcal{L}} \left(\sum_{s|v_s \in S} \sum_{j|\langle v_i,v_j\rangle \in E} \sum_{l|Q_l \in QS^m} L_{i,k}^{j,l}(s) + \sum_{s|v_s \in S} \sum_{j|\langle v_j,v_i\rangle \in E} \sum_{l|Q_l \in QS^m} L_{j,l}^{i,k}(s) \right) \leq 1 - \frac{1}{\mathcal{L}} \qquad \forall v_i \in V,\ \forall Q_k \in QS^m \tag{13}$$

$$\sum_{k|Q_k \in QS^m} P_i^k \leq 1 \qquad \forall v_i \in V \tag{14}$$

$$\sum_{s|v_s \in S} L_{i,k}^{j,k}(s) \leq C_k \qquad \forall \langle v_i, v_j\rangle \in E,\ \forall Q_k \in QS^m \tag{15}$$

$$\sum_{s \in S} L_{i,k}^{j,l}(s) \leq C_{k,l} \qquad \forall \langle v_i, v_j\rangle \in E,\ \forall Q_k, Q_l \in QS^m, k \neq l \tag{16}$$

Fig. 3. The proposed formulation for the QSR problem.

By solving the BIP, the flow routes and the nodes' quorum schedules will be jointly determined. Evidently, due to the conjoint nature of the solution, the resultant routes do not necessarily correspond to plain shortest routes; instead, the flow routes together with the nodes' chosen quorum schedule is to be determined with "activity ratio" minimization as the main objective.

4.3. A distributed solution to QSR problem

CQSR is computed as the solution of a BIP formulation, and as a result, its computational complexity can be prohibitive in large-scale scenarios. In this section, we provide a distributed near-optimal solution for the QSR problem in wireless sensor networks. We call this algorithm distributed quorum-based sleep scheduling and routing (DQSR for short).

In QSR problem, we seek to jointly determine the source-sink packet flow routes and the most efficient quorum (class) schedule of the nodes. A node's quorum schedule should ideally belong to a class with the least activity ratio, while still satisfying the number of overlapping awake slots needed by that node to successfully communicate with its neighbors. Therefore, we may think of this overlap number as the main factor in specifying a node's quorum schedule. When routing comes into play, the number of overlaps should be such that the rate required by the source nodes is satisfied. In DQSR, it is this notion of overlap that we will exploit to prepare the ground for the operation of a distributed graph-theoretic algorithm. Before delving into the details, we give a brief overview of the DQSR's main ideas:

Given an MCQS as the pool of the nodes' potential schedules, we first build a logical graph on top the network's topology which will encode the information regarding all potential number of overlaps between neighboring nodes. For easier reference, we henceforth call this new logical structure, the *overlap graph* and denote it by $G'' = \langle V'', E'' \rangle$. As before, let C^* be the maximum possible overlap size across all quorum classes within the given MCQS. Each node in the topological graph G is replicated C^* times (per each neighbor) in the overlap graph G''. The vertex set V'' of G'' is called the set of *overlap vertices* to avoid confusion with elements of V. Each overlap vertex corresponds to a particular overlap occurrence. Similarly, we may refer to the elements of E'' as the *overlap edges*. Each overlap edge is assigned a weight equal to the activity ratio of the quorum class which can support the required overlap size with minimum awake slots. We will show that by determining minimum-weight paths over G" as a sequence of overlap vertices, quorum schedules with minimum activity ratios will automatically be specified. Furthermore, since each overlap occurrence between two neighbors amount to a single transmission opportunity, each overlap vertex should lie only on a single path. Hence, the paths to be found in G'' should be vertex-disjoint. This will guarantee a capacity of one packet per frame for each overlap edge, and thus a data rate of one packet per frame for each vertex-disjoint path. This way, the total number of vertex-disjoint paths required is equivalent to the total number of packets needed to be sent by all source nodes within a single frame. With these preparations made, the distributed solution of the QSR problem essentially comes down to the execution of a minimum-weight vertex-disjoint paths algorithm over the overlap graph.

With this overview in mind, we may organize DQSR' operation into two phases: the first phase is dedicated to the construction of the overlap graph G''. In the second phase, we run a minimum-weight vertex-disjoint path algorithm over G'' to jointly determine the source-sink packet flow routes and the quorum schedule of the nodes. Our assumptions in this section are similar to those outlined in Section 4.1. In what follows, we discuss DQSR's phases in Sections 4.3.1 and 4.3.2.

4.3.1. Phase one: overlap graph construction

A node's quorum schedule depends on the number of overlapping awake slots it needs to successfully communicate with its neighbors. Hence, we construct the overlap graph based on maximum possible number of overlapping slots C^* between every pair of neighboring nodes. As already mentioned in Section 4.2.2, C^* is a property of the given MCQS instance. More specifically, consider two neighbor nodes v_i and v_j operating via an MCQS-based schedule. These two nodes may have at most C^* overlapping awake slots; i.e., within the duration of a frame, a node v_i can transmit at most C^* packets to each of its neighbors.

Now, consider a node v_i with d neighbors. Also, assume that v_i needs $x_1, \ldots, x_d$ overlaps with its neighbors. Obviously, v_i is better off by choosing its quorum schedule from a class $QC^* \in QS^m$

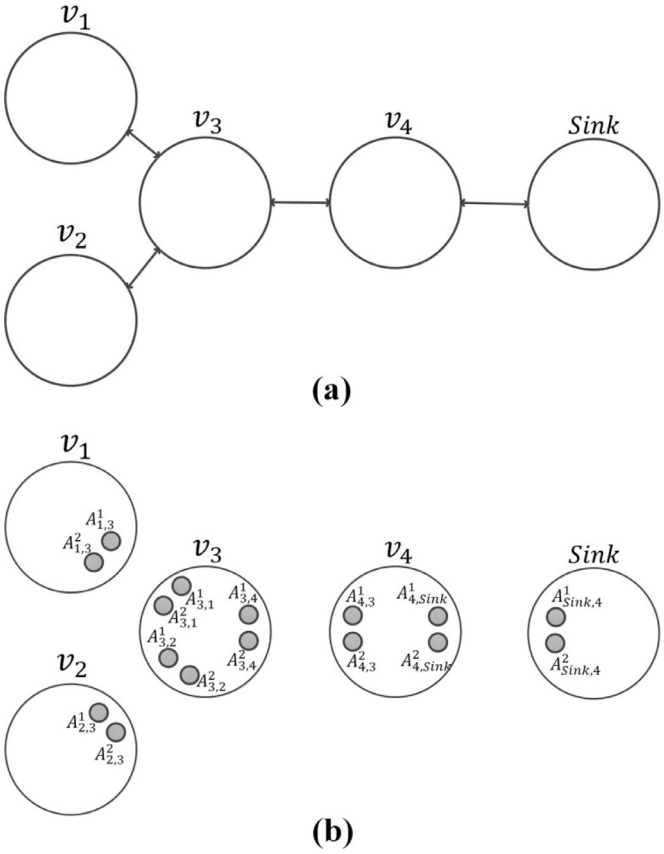

Fig. 4. Step 1 of overlap graph construction: determining overlap vertices. (a) Topological graph. (b) Overlap vertices in the overlap graph.

whose activity ratio is the smallest, but still guarantees to v_i its required number of overlapping awake slots $x = Max\ \{x_1, \ldots, x_d\}$. We denote the activity ratio of such QC^* by $LAR(x)$. Thus, we have:

$$LAR(x) = \min_{1 \le k \le m} \{AR(Q_k) | \forall l \in [1, \ldots, m] : C_k, C_{k,l} \ge x\}$$

Now, we are ready to explain how G'' can be constructed from G:

- **Step 1)** For each node $v_i \in V$, and for each of its neighbors v_j, a total of C^* overlap vertices will be created in G". Each overlap vertex is designated by $A^a_{i,j}$, where $1 \le a \le C^*$ is used to index the overlap occurrences between v_i, v_j. More specifically, $A^a_{i,j}$ denotes the a-th overlap of node v_i with its neighbor v_j; i.e., $A^a_{i,j} \in V''\ if\ \langle v_i, v_j \rangle \in E, a \in [1, \cdots, C^*]$. As an example, in Fig. 4(a), a topological graph with four sensor nodes is depicted. Fig. 4(b) shows the overlap vertices in the corresponding overlap graph. Note that the figures are drawn under the assumption that the underlying MCQS system is such that $C^* = 2$.
- **Step 2)** For every pair of neighboring nodes v_i and v_j, each overlap vertex $A^a_{i,j}, a = 1, \ldots, C^*$ will be connected to the overlap vertex $A^a_{j,i}$ via a directed link. In other terms, $\langle A^a_{i,j}, A^a_{j,i} \rangle \in E''\ if\ \langle v_i, v_j \rangle \in E, a \in [1, \ldots C^*]$. The weight assigned to the directed link $\langle A^a_{i,j}, A^a_{j,i} \rangle$ is equal to $LAR(a)$. In fact, when finding the vertex-disjoint paths in phase 2, if it turns out that two neighbor nodes need a overlaps, their connecting edge in G" will have a weight of $LAR(a)$. Fig. 5 illustrates the changes made on G" during this step.
- **Step 3)** For all $v_i \in V$, we connect each overlap vertex $A^a_{i,j} \in V'', j | \langle v_i, v_j \rangle \in E, a \in [1, \ldots, C^*]$ to each overlap vertex $A^b_{i,u} \in V'',\ u | \langle v_u, v_i \rangle \in E, b \in [1, \ldots, C^*]$ if v_i and v_u are neighbors of v_i. In other words, $\langle A^a_{i,j}, A^b_{i,u} \rangle \in E''\ if\ \langle v_i, v_j \rangle \in E,\ \langle v_u, v_i \rangle \in E,\ a, b \in [1, \ldots, C^*]$. These edges are added only to guarantee connectivity; thus, their weight is set equal to zero. Fig. 6 depicts the changes made on G" during this step.
- **Step 4)** G" is augmented with a new vertex, namely *super sink*, which we designate by Z (see Fig. 7). This vertex is in fact the endpoint of the overlap graph when using the vertex-disjoint paths algorithm of phase 2. Each overlap vertex of the sink will be connected to Z via a directed edge.
  In other words, $\langle A^a_{sink,j}, Z \rangle \in E'', j | \langle v_j,\ sink \rangle \in E, a \in [1, \ldots, C^*]$. These edges are dummies and are there only to prepare the ground for the operation of the vertex-disjoint paths algorithm; thus, their weight is set equal to zero.
- **Step 5)** Consider the source node v_s. Let $p = 1, \ldots, R_s$ denote the index of each of the R_s packets v_s intends to send during a

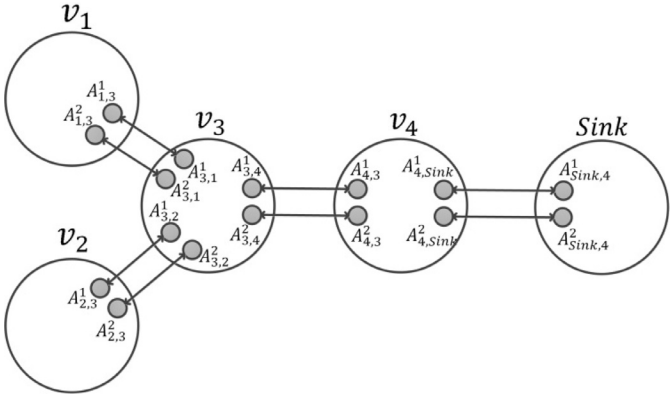

Fig. 5. Step 2 of overlap graph construction: adding inter-node edges.

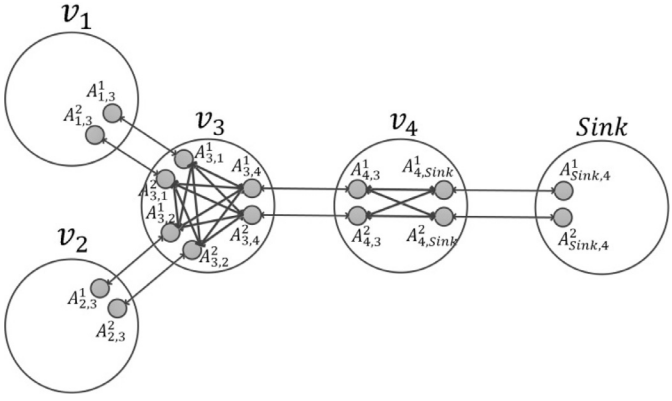

Fig. 6. Step 3 of overlap graph construction: adding intra-node edges.

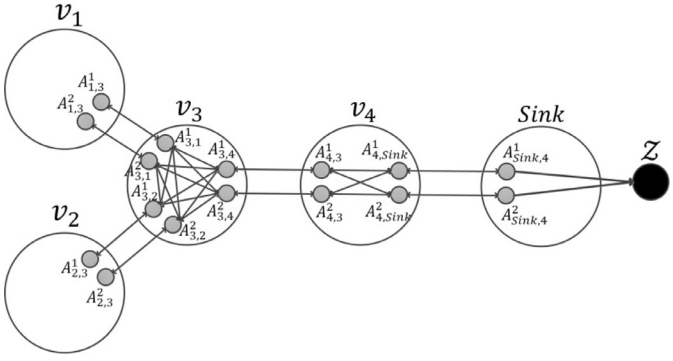

Fig. 7. Step 4 of overlap graph construction: augmenting the overlap graph with a super sink.

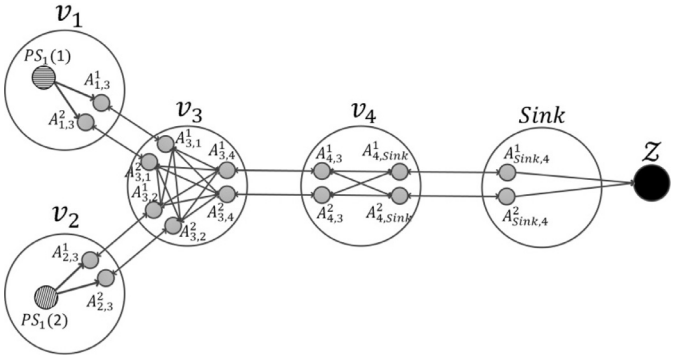

Fig. 8. Step 5 of overlap graph construction: augmenting the overlap graph with pseudo sources.

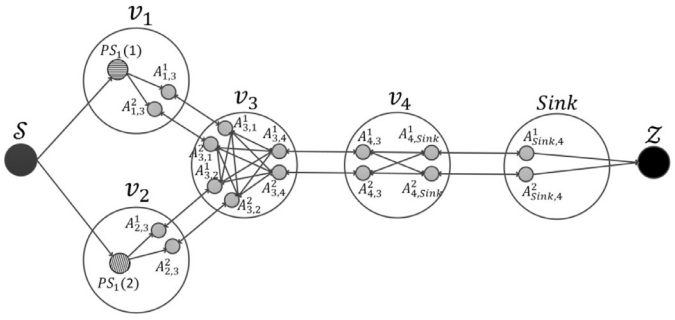

Fig. 9. The last step of overlap graph construction: augmenting the overlap graph with super source.

time frame. For all such sources v_s and for each of its intended packets p, we introduce a new vertex $PS_p(s)$ called a *pseudo source*. This is shown in Fig. 8. Augmentation of G" with these vertices is necessary to ensure that sufficient vertex-disjoint paths would exist to support all the packets issued from all the sources. Each pseudo source $PS_p(s)$ is then connected to all the overlap vertices of v_s, again via zero-weight directed edges.

- **Step 6)** G" is augmented with a new vertex, namely *super source*, which we designate by S. This vertex is in fact the starting point of the overlap graph when using the vertex-disjoint paths algorithm of phase 2. S will be connected to each overlap vertex within a pseudo-source via a zero-weight directed edge. In other words, $\langle S, PA_p(s)\rangle \in E'', s|v_s \in S, p, \in [1, \ldots, R_s]$.

In Fig. 9, we illustrate the complete overlap graph corresponding to the topology in Fig. 4(a).

4.3.2. Phase two: finding vertex-disjoint paths in overlap graph

Let $M = \sum_{s|v_s \in S} R_s$ be the total number of packets to be sent during a frame from different source nodes to the sink; hence, M is also equal to the number of pseudo-source vertices in the overlap graph. Now, one can use the standard minimum-weight vertex-disjoint paths algorithm to find M vertex-disjoint paths between the super-source PS and super sink Z. Each vertex-disjoint path goes through one of the pseudo-source nodes, and represents the packet route corresponding to that pseudo-source vertex. Rather than bore the reader with the detailed implementation of this path finding procedure, we defer it to Appendix A, where we also hint at how these paths can also be calculated in a distributed fashion. For now, assume that the desirable vertex-disjoint paths have been computed. Consider the source node $v_s \in S$. DQSR determines the v_s-to-*sink* route for each of the R_s packets of v_s. In particular, the node v_i lies on the p-th v_s-to-*sink* path if one of the v_i's overlap vertices lies on the vertex-disjoint path which starts from super source S, passes through the pseudo-source $PS_p(s)$, and ends in super sink Z. Also, once DQSR terminates, each node can determine its required number of overlaps with its neighbors. More specifically, consider two neighbors v_i and v_j. Let $O = \{A^a_{i,j},\ a = 1, \cdots, C^*\}$ denote all v_i's overlap vertices defined in association with its neighbor v_j. Let $O' \subseteq O$ be that subset of v_i's overlap vertices which happen to lie on one of DQSR's determined vertex-disjoint paths. Then, $x_j = max\{a : A^a_{i,j} \in O'\}$ would be the required number of overlaps between v_i and its neighbor v_j. With the required number of overlaps x_j at hand, v_i then picks its schedule from a quorum class which can provide for x_j while having the minimum activity ratio. More precisely, assume that v_i has d neighbors. Also, assume that v_i needs $x_1, \ldots, x_d$ overlaps with these neighbors. If $x = max\{x_1, \ldots, x_d\}$, then v_i picks its schedule from a class which has $LAR(x)$ as its activity ratio.

4.3.3. Computational complexity

In DQSR, we have to find a total of $M = \sum_{s|v_s \in S} R_s$ minimum-weight node-disjoint paths between the super-source and super-sink in the overlap graph $G'' = \langle V'', E''\rangle$. As we have discussed in the Appendix, this is done in M steps, and in each step, exactly one node-disjoint path is found. In [30], an algorithm has been proposed for finding a minimum-weight node-disjoint path whose time complexity is $O(m.\log_{1+\frac{m}{n}} n)$, where n is the number of vertices, and m is the number of edges in the given graph. Also, as argued in [30], finding M minimum-weight node-disjoint paths can be accomplished with complexity $O(M.m.\log_{1+\frac{m}{n}} n)$. On this basis, it is straightforward to compute the time complexity of our DQSR algorithm; it is of the order $O(M.|E''|.\log_{1+\frac{|E''|}{|V''|}} |E''|)$, in which:

$$\left|V''\right| = 2|E|.C^* + M + 2 \tag{17}$$

$$\left|E''\right| = C^*.|E| + \left[\sum_{i|v_i \in V - \{sink\}} b_i(b_i - 1)\right].C^{*2} + C^* + M \tag{18}$$

In (18), b_i denotes the neighborhood degree of node v_i in the topological graph of the network.

5. Performance evaluation

In this section, we evaluate several instances of our proposed MCQS construct introduced in Section 3. We then go on to investigate the performance of the DQSR algorithm of Section 4.3 for the QSR problem. We organize this section as follows: In 5.1, we study the impact of different system parameters on MCQS. We also compare MCQS with some well-known quorum systems from the literature. In 5.2, we evaluate DQSR. To this end, DQSR is compared with CQSR (i.e., our theoretical optimal solution for the QSR problem) as well as with the HQS-based sleep scheduling algorithm [22] in terms of activity ratio, energy consumption, and flow latencies. We also explore the impact of the number of MCQS classes and the number of flows on DQSR's performance.

We have implemented our BIP models in AIMMS v. 3.11 [33]. These models are then solved using CPLEX v. 12.3 [34]. For comparison between CQSR, DQSR, and HQS-based sleep scheduling, we use network simulations (See Section 5.2 for further details).

5.1. Evaluating MCQS

In this section, we first explore the impact of MCQS parameters on its performance. We then compare MCQS with some of the well-known quorum systems in terms of activity ratio. Since in MCQS all quorums belonging to a given class are identical in terms

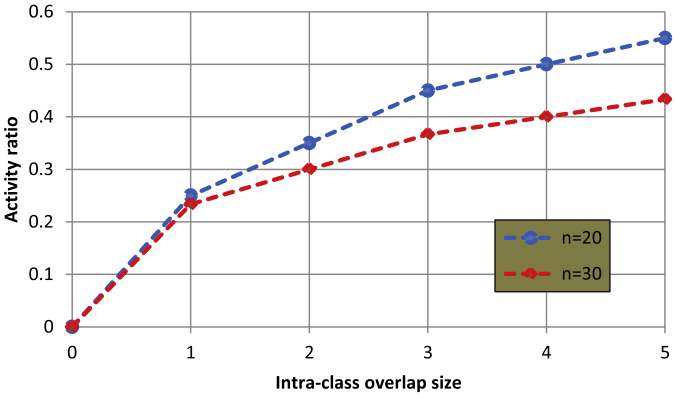

Fig. 10. Impact of intra-class overlap size.

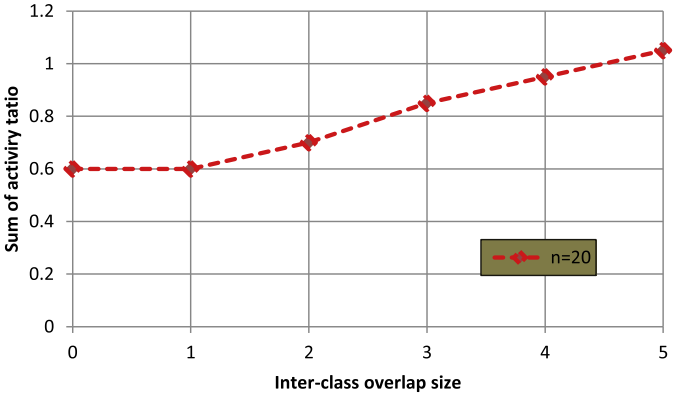

Fig. 11. Impact of inter-class overlap size.

of their activity ratio and guaranteed overlap size, each node is free to choose any one of the quorums from the class. Accordingly, in our experiments, we assume that each class contains only one quorum (i.e., $QC_i^n = \{Q_i\}$). We lose no generality by this simplification, as one can easily derive other elements of QC_i^n by appropriately rotating Q_i.

5.1.1. Exploring the impact of MCQS parameters

Specializing MCQS for different applications is done through its parameters. In this section, we explore the impact of MCQS's cycle length n, its intra-class overlap size C_i (for each class i), and its inter-class overlap size $C_{i,j}$ (for each pair of classes i and j).

- **Impact of the intra- and inter-class overlap size**
  The most important criterion for evaluating quorum systems is activity ratio. A given quorum's activity ratio is computed as the ratio of its cardinality to the system's cycle length. However, for a fixed system cycle length, quorum cardinality is determined by the guaranteed overlap size. Hence, the activity ratio is dependent on the guaranteed overlap size. Fig. 10 shows the impact of the intra-class overlap size on the quorum's activity ratio in a single class quorum system.
  In Fig. 11, we show the impact of the inter-class overlap size on sum of the activity ratios of the quorums in an MCQS with $m = 2$ classes and $C_1 = C_2 = 1$. The Y-axis now represents the sum of the activity ratios for these two classes.
  As can be seen in Figs. 10 and 11, in an MCQS with a fixed cycle length, the activity ratio increases with the intra- and inter-class overlap size.
- **Impact of the cycle length**
  Activity ratio is also affected by the system's cycle length. In Fig. 12, we show the impact of cycle length on the quorum's activity ratio in a single-class quorum system. Results are also given for different intra-class overlap sizes ($C = 1, 2, 3$).

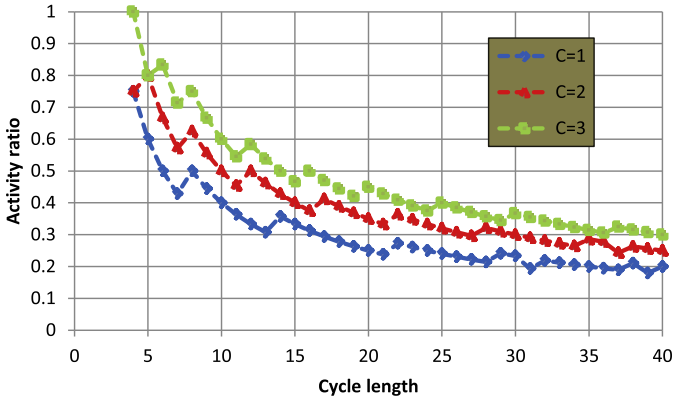

Fig. 12. Impact of cycle length.

As can be seen in Fig. 12, in general, for a specific guaranteed overlap size, the quorum activity ratio decreases with system's cycle length. Exceptions can also be seen in some points where the activity ratio has increased with cycle length. For instance, consider the case with $C = 1$ in Fig. 12. Activity ratio has spiked immediately after cycle lengths 7, 13, 21, 28, 31, 37, and 39. These points can essentially be considered as local minima, and one may use these as efficient cycle lengths to construct a more energy-aware system.

On the other hand, in quorum-based sleep-scheduling algorithms, cycle length amounts to the maximum number of time slots until two neighboring nodes rendezvous. Therefore, increasing the cycle length, while reducing the activity ratio, would also result in inferior neighbor sensibility. MCQS's main advantage lies in the fact that the system's cycle length can be adjusted arbitrarily.

5.1.2. MCQS Vs. other quorum systems

Existing quorum systems have different overlapping and activity ratio properties. Also, in some cases, the system's cycle length is limited to specific numbers. MCQS is a generalization of quorum systems with a set of tunable parameters which can be exploited to instantiate a system similar to almost all existing quorum systems. In this section, in order to contrast MCQS with other quorum systems, we specialize it in each case to derive a comparable construct. The most important criterion for comparison of quorum systems is their activity ratio. However, as already mentioned before, the activity ratio, in turn, depends on the guaranteed overlap size. For a fixed cycle length, the larger the guaranteed overlap size, the larger the quorum activity ratios would become. Each of the proposed quorum systems in the literature guarantees a certain overlap size, but in MCQS, we may tune the guaranteed overlap size with some flexibility. Therefore, to compare MCQS with other systems in terms of activity ratio, it suffices to consider an MCQS instance with fixed cycle length and then set the guaranteed overlap size as that of the comparable system.

Perhaps the most well-known quorum construct in the literature is the grid quorum system. In this system, all quorums are homogeneous with a guaranteed overlap size of 2. We compare this system with an MCQS of $m = 1$ and $C_1 = 2$. As can be seen in Fig. 13, for all system cycle lengths, MCQS has consistently smaller activity ratio. Of particular note is that unlike the grid system which is limited to square cycle lengths, MCQS can be constructed for all system cycle lengths.

Cyclic quorum system is another standard quorum construct. This system comes with a single overlap guarantee. We compare this system with an MCQS of $m = 1$ and $C_1 = 1$ (see Fig. 14). It

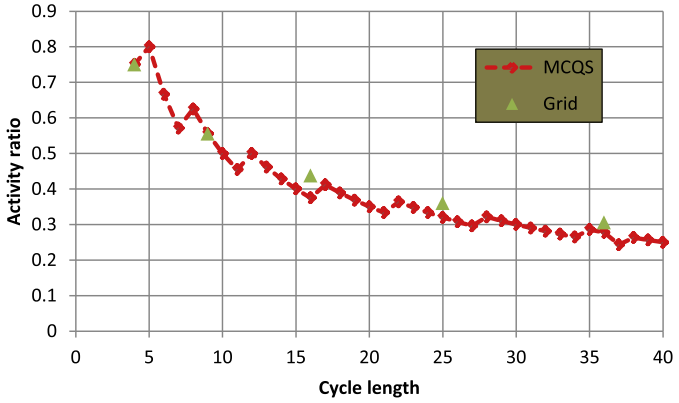

Fig. 13. Comparison of MCQS with grid quorum system.

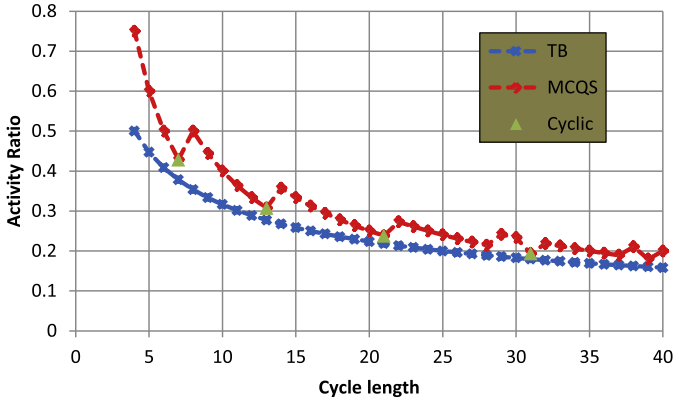

Fig. 14. Comparison of MCQS with cyclic quorum system and theoretical benchmark.

is shown in [13] that the optimal bound on the activity ratio of a quorum system with cycle length n and with a single overlap guarantee is $\sqrt{n}/n$. Hence, the cyclic quorum system is, in fact, an instance of the optimal quorum construct. However, unlike MCQS, the cyclic system can only be instantiated for specific cycle lengths. For instance, in Fig. 14, the cyclic system's existence is limited to cycle lengths of 7, 13, 21, and 31. The figure also depicts the theoretical bound (TB) for completeness. As expected, MCQS's activity ratio turns out to be identical to that of the cyclic system. The slight gap between MCQS and the theoretical bound is due to the fact that the quorum size should be an integer.

Next, we compare MCQS with the e-torus system which is primarily used in adaptive sleep-scheduling algorithms. A quorum system consisting of e-torus(1) and e-torus(2) can be compared with an MCQS of $m = 2$, $C_1 = 1$, $C_2 = 2$, and $C_{1,2} = 1$. In fact, e-torus(1) and e-torus(2) correspond to QC_1 and QC_2, respectively. In Fig. 15, we plot the activity ratios of these four constructs for varying cycle lengths. As can be seen, for any system cycle length, the activity ratios of QC_1 and QC_2 is at most equal to those of e-torus(1) and e-torus(2), respectively.

In our next set of experiments, we compare MCQS against HQS [22] in terms of activity ratio. HQS is used in adaptive sleep-scheduling algorithms. For the sake of this comparison, we first need to put these two quorum systems in the same context: We use a $(n_1 = n/2,\ n_2 = n;\ \phi_1 + n - 1)$-HQS (See [22]). Accordingly, we create an MCQS with cycle length n (to be varied in evaluations) and with other parameters chosen as: $m = 2$, $C_1 = 1$, $C_2 = 2$, and $C_{1,2} = 1$. In an HQS configured this way, a quorum with cycle length n_1 corresponds to QC_2 and a quorum with cycle length n_2 corresponds to QC_1. In other words, in such HQS, in every n

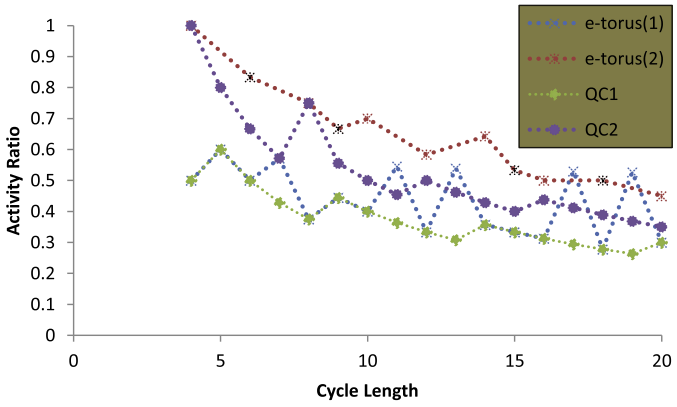

Fig. 15. Comparison of MCQS with e-torus.

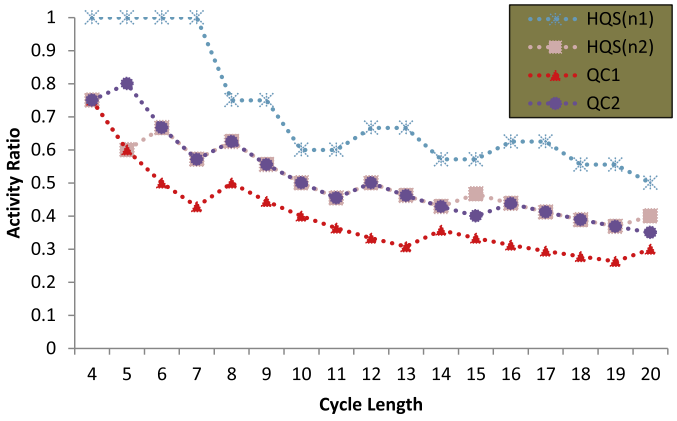

Fig. 16. Comparison of MCQS with HQS.

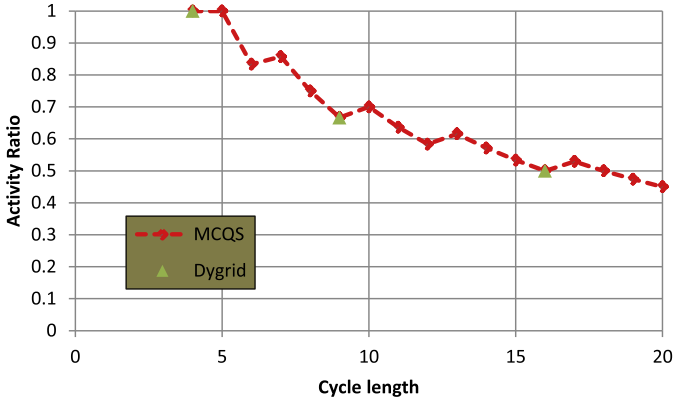

Fig. 17. Comparison of MCQS with Dygrid quorum system.

consecutive slots, it is guaranteed to have two overlaps between any two quorums with cycle length n_1, and to have one overlap between any two quorums with cycle length n_2. Also, during every $\phi_1 + n - 1$ consecutive slots, it is made sure that any n_1-sized quorum has one overlap with any n_2-sized quorum. This roughly corresponds to the overlap between the classes QC_1 and QC_2 in MCQS. In Fig. 16, we compare the activity ratios of (n_1), $HQS(n_2)$, QC_1 and QC_2 for different cycle lengths.

Finally, we choose Dygrid as a representative of quorum systems used in asymmetric sleep-scheduling algorithms. As pointed out earlier in Section 2, *Dygrid*$(r,\ c,\ k_1,\ k_2)$ is an optimal quorum system. We compare this system with an MCQS of $m = 2$, $C_1 = C_2 = 0$ and $C_{1,2} = k_1 \times k_2$. In Fig. 17, we show the activity ratio of these two systems. We have assumed, $k_1 = k_2 = 1$ for this

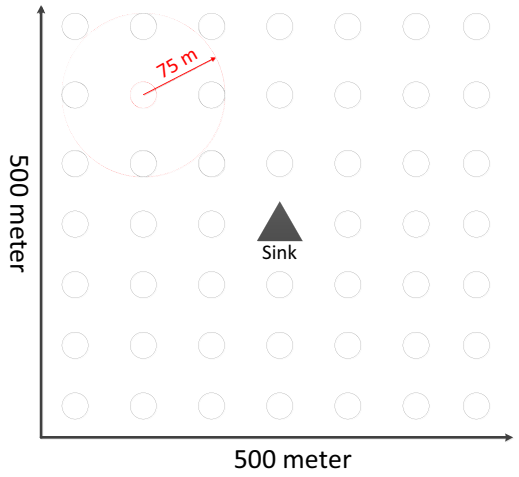

Fig. 18. Distribution of the nodes in the network area.

Table 2
Simulation parameters.

Parameter	Value
Number of nodes	50
Network area	500 m × 500 m grid
Transmission range	75 m
Channel capacity	250 Kbps
Packet size	256 Bytes
Energy consumption model	MICAz
Transmit power	52.2 mW
Receive power	83.1 mW
Idle power	105 µW
Sleep power	48 µW
Initial energy	10 J
Time slot duration	20 ms
TDMA frame length	30 slots
Number of classes in MCQS instance	4
Source rate	1 packet/frame
Simulation duration	600 s

experiment. As can be seen, similarly to Dygrid, MCQS enjoys from an optimal activity ratio. This is while the Dygrid's existence is restricted to square cycle lengths.

An important observation in obtained results is the lower activity ratio of MCQS compared to existing quorum systems. As we also discussed in Section 3.2.3, this gain in terms of activity ratio (together with its more general design) is achieved by paying a higher computational cost for calculating an optimal MCQS. As argued in Section 3.2.3, an optimal activity ratio is well worth the computational cost, especially given that this cost is affordable in typical sleep scheduling scenarios and that MCQS is a one-time computation job.

5.2. Evaluating DQSR

In Section 4.2, the QSR problem has been formulated as a BIP model. The optimal solution to QSR (referred to as CQSR) has a minimum activity ratio. In Section 4.3, we also proposed a distributed solution to the QSR problem, namely the DQSR algorithm. In this section, we compare DQSR with CQSR (as a benchmark). We also compare DQSR with HQS-based sleep scheduling algorithm [22]. In this algorithm, the quorum schedules (with different cycle lengths) are chosen according to the nodes' traffic load. The DQSR's performance is contrasted against CQSR and HQS-based algorithm in terms of nodes' energy consumption and flow latencies. In addition, we investigate the impact of the number of MCQS classes on both CQSR and DQSR.

To implement DQSR, we use MATLAB. In particular, given the knowledge of the network topology and the source nodes, we compute the nodes' sleep schedules and flow routes for each test scenario. We then apply the results to the MAC and network components of the NS-2 simulator [35]. More specifically, the flow routes have been applied using a simple source routing scheme, and the sleep/wake-up patterns of the nodes are adjusted according to the computed quorum schedules.

We conduct the experiments for a network of 50 nodes. It is assumed that the nodes are laid out in a $500\,m \times 500\,m$ grid-like structure with the sink node located approximately at the center (Fig. 18). The transmission range of a sensor node is 75 m. The channel capacity is 250 Kbps. The packet size is considered to be 256 bytes. The energy consumption model of MICAz [32] is employed in the simulation, where the power consumption for transmit, receive, idle, and sleep modes are 52.2 mW, 83.1 mW, 105 µW, and 48 µW, respectively. Each sensor node has an initial energy of 10 J. A time slot is set to be 20 ms long. We assume a frame length of 30 slots. Therefore, the nodes' quorum schedules should have a cycle length of 30. Unless otherwise stated, the MCQS system used throughout the experiments has a cycle length of 30 and has a total of 4 quorum classes with just a single quorum within each class. In particular, we use the following MCQS construct as the basis for our experiments:

$$QS_4^{30} = \left\{QC_1^{30} = \{\emptyset\}, QC_2^{30} = \left\{\{0, 1, 5, 9, 13, 15, 16\}\right\},\right.$$
$$\left.QC_3^{30} = \left\{\{0, 1, 9, 15, 16, 17, 21, 25, 28\}\right\}, QC_4^{30} = \{U\}\right\}$$

The quorum in class QC_1^{30} indicates a fully sleep status across all slots, and the single quorum within QC_4^{30} indicates a fully-awake status across all slots. The activity ratios of the quorums in these classes are 0, 7/30, 9/30, and 1, respectively.

For fair comparison of DQSR with HQS-based sleep scheduling algorithm, we use a $(n_1 = 15,\ n_2 = 30; 32)$-HQS. In other words, in this HQS, two quorums G_1 and G_2 are used whose sizes are $n_1 = 15$ and $n_2 = 30$, respectively. For example, any two nodes adopting G_1 as their quorum schedule are guaranteed to have two overlaps within a 30-slot frame. This corresponds to the case of QC_2. In addition to G_1 and G_2, we also use two additional sleep/wake-up patterns in HQS: a G_0 (Fully Sleep) and a G_3 (Fully Awake) pattern (this is to make it comparable with CQSR and DQSR). As such, the nodes are free to choose one of the following patterns as their schedules:

$$G_0 = \{\emptyset\},\ G_1 = \{0,\ 1,\ 2,\ 3,\ 6,\ 9,\ 12\},$$
$$G_2 = \{0,\ 1,\ 2,\ 3,\ 4,\ 8,\ 12,\ 16,\ 20,\ 24\},\ G_3 = \{U\}$$

Moreover, while in DQSR, the nodes' quorum schedules as well as the flow routes are "jointly" determined together, there is no discussion in [22] as to what routing mechanism is to be used. For comparison, we have used a shortest-path algorithm along with HQS-based sleep scheduling for determining the flow routes.

We conduct the experiments for different numbers of source nodes (flows). The source nodes are chosen randomly, and have a transmission rate of 1 packet per each time frame. Each simulation run lasts 600 s (1000 frames). Each data point in the subsequent plots shows an average of 10 simulation runs. For each point in the results, its 90% confidence interval has also been given. Important simulation parameters are listed Table 2.

- **Impact on energy consumption**
  In Fig. 19, we compare DQSR with CQSR and HQS-based sleep scheduling algorithm in terms of the nodes' energy consumption. We conduct this comparison for different percentages of source nodes where simulations run for 600 s. As a general trend in Fig. 19, as the number of flows increases, the nodes would experience heavier load. This in turn calls for quorum

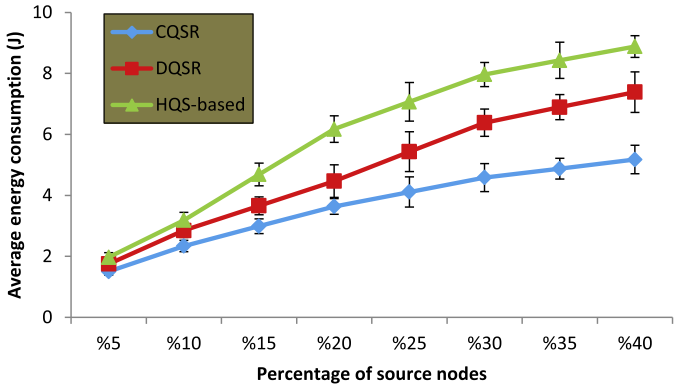

Fig. 19. Impact of the number of flows on energy consumption.

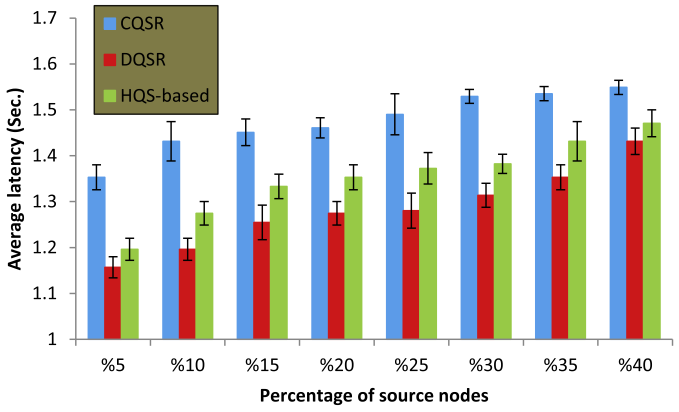

Fig. 20. Impact of the number of flows on average delay.

schedules with a larger guaranteed overlap size so that the nodes can keep up relaying the packets to their neighbors. Hence, the larger the number of flows, the larger the nodes' energy consumption. Among the three schemes, CQSR (i.e. the optimal solution for the QSR problem) has the lowest energy consumption. Under light traffic conditions, DQSR and the HQS-based algorithm perform similarly. However, as the traffic load increases, DQSR maintains a smaller margin with respect to the optimal bound determined by CQSR. DQSR's superior performance is due to two facts: Firstly, in DQSR, the flow routes and the nodes' quorum schedules are determined "jointly". As such, the traffic load would be distributed in a way that leads to a lower total energy consumption. This is while the HQS-based scheme always chooses the shortest paths, which leads to heavy energy usage by the nodes lying along these paths. Secondly, the MCQS-based quorums have a smaller activity ratio compared to HQS. This in turn leads to more energy saving in DQSR.

- **Impact on flow latencies**
  In Fig. 20, we compare DQSR, CQSR and HQS-based sleep scheduling algorithm with respect to the flow latencies. As it turns out, the latency induced by DQSR is less than that of CQSR. This can be attributed to the fact that CQSR is only optimal in terms of activity ratio, and our objective function in (8) does not explicitly account for flow latencies. In fact, as evidenced in this experiment, DQSR's suboptimal (and thus consistently higher) activity ratio can in turn lead to higher neighbor sensibility and thus smaller flow latencies.
  Also, as can be seen in Fig. 20, the flow latencies in DQSR is slightly lower compared to the HQS-based algorithm. This is despite the fact that the HQS-based algorithm always chooses the shortest paths for the flows. In fact, while the number hops traversed by each flow is smaller in the HQS-based scheme, the latency between any two neighbors can become higher compared to the case in DQSR. In other terms, the nodes along the shortest paths would be more heavily burdened in the HQS-based scheme, which can give rise to a higher per-hop latency. From Fig. 20, we also observe the impact of the number of source nodes on the average delay induced by DQSR, CQSR and the HQS-based algorithm. As can be noted, the flow latencies do not significantly change (magnitude-wise) since all the three solutions can accommodate the resultant increase in traffic load by utilizing quorums with higher activity ratios.

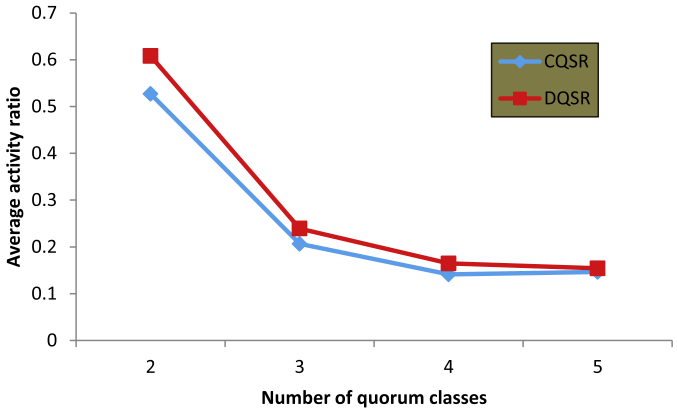

Fig. 21. Impact of the number of quorum classes on activity ratio.

- **Impact of the number of MCQS classes on activity ratio**
  In QSR problem, the quorum (class) schedule for each node is governed by the traffic load on that node. Obviously, the search space for the nodes' quorum schedules grows larger as we increase the cardinality of the MCQS construct. In Fig. 21, we demonstrate the impact of the number of quorum classes on the nodes' activity ratios in both CQSR and DQSR. We use MCQS instances with 2 to 5 classes. Each MCQS instance includes a class with zero activity ratio (totally sleep schedule) and a class with activity ratio of 1 (totally awake schedule). In general, the average activity ratio decreases when an MCQS with a larger cardinality is utilized. However, the slope of increase eventually evens out, and the average activity ratio remains constant. From this experiment, we can conclude that beyond a certain threshold, no gain can be obtained by enlarging the search space to include classes with a higher intra- and inter-class overlap size.

6. Conclusion

We have introduced a multi-class quorum system (MCQS) which generalizes the classical definition of quorum systems. The activity ratio of quorums within a given class in MCQS is different from that of quorums in other classes. Also, the intra- and inter-class overlap size may differ across the classes of a given MCQS. We have shown how the parameters of an MCQS construct can be leveraged to specialize it for use in different sleep-scheduling paradigms. We have also extended the standard rotation closure property of quorum systems in order for MCQS to become applicable to asynchronous sleep-scheduling algorithms. In this article, MCQS has been compared with some of the well-known quorum systems from the literature. It has been shown that MCQS is optimal in terms of activity ratio regardless of its parametric configuration. Moreover, unlike some standard quorum systems, MCQS does not impose any limitation on either the system's cycle length or on achievable guaranteed overlap size.

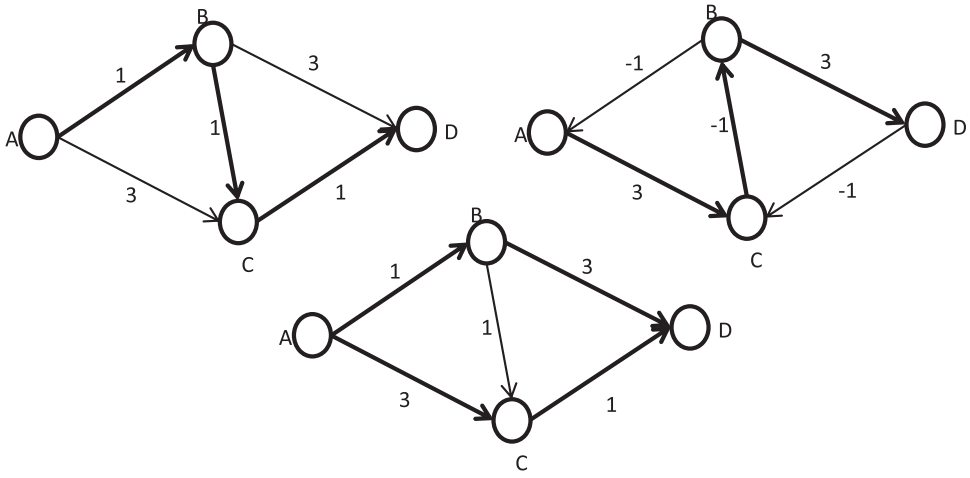

Fig. 22. Two minimum weight vertex-disjoint paths from A to D.

We have also presented a case study to illustrate the application of MCQS to sleep scheduling in wireless systems. In particular, we leverage on both MCQS and cross-layering to present a BIP model for the joint problem of quorum-based sleep-scheduling and routing (QSR) in wireless sensor networks. We have come up with novel centralized and distributed solutions for the QSR problem. As evidenced by the simulation experiments, our distributed solution, while computationally more efficient, is capable of achieving near-optimal performance.

Appendix A. Minimum weight *M*-vertex-disjoint paths algorithm

In this section, we give a brief overview of the minimum weight M-vertex-disjoint paths algorithm which is used in the second phase of the DQSR algorithm.

Consider a weighted graph $G = (V, E)$. Let $w_{u,v}$ denote the weight of the edge $\langle u, v\rangle \in E$. The problem of finding minimum weight M-vertex-disjoint paths on graph G is defined as follows:

Find M vertex-disjoint paths $(P_1, P_2, \ldots, P_M)$ from a source vertex A to a destination vertex D such that the sum of the weights of all the paths is minimized; i.e.,

$$Minimize\ Z = \sum_{i=1}^{M} W(P_i)$$

where, $W(P_i)$ denotes the weight of path P_i. More precisely, $W(P_i)$ is equal to the sum of the weights of the edges forming path P_i:

$$W(P_i) = \sum_{\langle u,v\rangle \in P_i} w_{u,v}$$

An algorithm for finding the minimum-weight vertex-disjoint paths can use any standard shortest-path procedure (e.g., Bellman-Ford, Dijkstra, etc.) to find an initial vertex-disjoint path from A to D with minimum weight. To find the subsequent paths, some rearrangements need to be done on the graph. Let P_M denote a set containing M minimum weight vertex-disjoint paths. The $(M+1)$-st minimum weight vertex-disjoint path is then computed as follows: first, the direction of each edge along the paths in P_M is reversed and its weight is negated. We refer to these edges as negative edges, and to the rest as positive edges.

Each vertex v, on one of the paths in P_M, is split into two vertices v_1 and v_2. These two new vertices are then connected together with a zero-weight edge. The outgoing edges of v now exist from v_2, and the incoming edges of v now enter v_1.

Now, using Bellman-Ford or Dijkstra algorithms, the shortest path from A to D is computed in the modified graph. Denote this path by P', which may include both positive and negative edges.

By adding positive edges of P' to P_M, and also removing the negative edges of P' from P_M, a new set containing $M+1$ minimum weight vertex-disjoint paths is constructed. Denote this new set by P_{M+1}. In Fig. 22, we depict a simple example. In this figure, two minimum weight vertex-disjoint paths from A to D has been found.

In [29–31], several algorithms have been proposed which can find minimum weight M vertex-disjoint paths in a distributed fashion.

We may use these algorithms to find a total of $M = \sum_{s|v_s \in S} R_s$ minimum weight vertex-disjoint paths in our overlap graph $G'' = (V'', E'')$. Recall that in overlap graph, the source A is the super source vertex, and the destination D is the super sink vertex.

References

[1] C. Chih-Min, L. Yi-Wei, A quorum-based energy-saving MAC protocol design for wireless sensor networks,", Veh. Technol. IEEE Trans. 59 (2010) 813–822.
[2] W. Shan-Hung, C. Chung-Min, C. Ming-Syan, Collaborative wakeup in clustered ad hoc networks,", Sel. Areas Commun. IEEE J. 29 (2011) 1585–1594.
[3] C. Zi-Tsan, L. Yu-Hsiang, J. Rong-Hong, Optimal asymmetric and maximized adaptive power management protocols for clustered ad hoc wireless networks, Parallel Distrib. Syst. IEEE Trans. 22 (2011) 1961–1968.
[4] Y.-C. Kuo, Quorum-based power-saving multicast protocols in the asynchronous ad hoc network, Comput. Netw. 54 (2010) 1911–1922.
[5] C. Zi-Tsan, L. Yu-Hsiang, S. Tsang-Ling, Asynchronous power management protocols with minimum duty cycle and maximum adaptiveness for multihop ad hoc networks, Veh. Technol. IEEE Trans. 62 (2013) 3301–3314.
[6] C. Bong Jun, S. Xuemin, Adaptive asynchronous sleep scheduling protocols for delay tolerant networks, Mobile Comput. IEEE Trans. 10 (2011) 1283–1296.
[7] S. Singh, C.S. Raghavendra, PAMAS - power aware multi-access protocol with signalling for ad hoc networks, Comput. Commun. Rev. 28 (1998) 5–25.
[8] C.F. Chiasserini, R.R. Rao, A distributed power management policy for wireless ad hoc networks, in: Wireless Communications and Networking Conference, 2000. WCNC. 2000, vol.3, IEEE, 2000, pp. 1209–1213.
[9] IEEE Standard for Information Technology – Telecommunications and Information Exchange Between Systems – Local and Metropolitan Area Networks – Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11–2007 (Revision of IEEE Std 802.11–1999), pp. 1–1076, 2007.
[10] Y.C. Tseng, C.S. Hsu, T.Y. Hsieh, Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks, in: Proceedings IEEE INFOCOM, 2002, pp. 200–209.
[11] Y.C. Tseng, C.S. Hsu, T.Y. Hsieh, Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks, Comput. Netw. 43 (2003) 317–337.
[12] R. Zheng, J.C. Hou, L. Sha, Asynchronous wakeup for ad hoc networks, in: Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 2003, pp. 35–45.
[13] J.R. Jiang, Y.C. Tseng, C.S. Hsu, T.H. Lai, Quorum-based asynchronous power-saving protocols for IEEE 802.11 ad hoc networks, Mobile Netw. Appl. 10 (2005) 169–181.

[14] S. Lai, B. Ravindran, H. Cho, Heterogenous quorum-based wake-up scheduling in wireless sensor networks, IEEE Trans. Comput. 59 (2010) 1562–1575.
[15] Z.T. Chou, Optimal adaptive power management protocols for asynchronous wireless ad hoc networks, in: *IEEE Wireless Communications and Networking Conference, WCNC*, 2007, pp. 61–65.
[16] G. Ekbatanifard, R. Monsefi, M.H. Yaghmaee M, S.A. Hosseini S, Queen-MAC: a quorum-based energy-efficient medium access control protocol for wireless sensor networks, Comput. Netw. 56 (2012) 2221–2236.
[17] C.M. Chao, J.P. Sheu, I.C. Chou, An adaptive quorum-based energy conserving protocol for IEEE 802.11 ad hoc networks, IEEE Trans. Mobile Comput. 5 (2006) 560–570.
[18] M. Maekawa, Root n algorithm for mutual exclusion in decentralized systems, ACM Trans. Comput. Syst. 3 (1985) 145–159.
[19] R. Zheng, J.C. Hou, L. Sha, Optimal block design for asynchronous wake-up schedules and its applications in multihop wireless networks, IEEE Trans. Mobile Comput. 5 (2006) 1228–1240.
[20] W. Shan-Hung, C. Chung-Min, C. Ming-Syan, An asymmetric quorum-based power saving protocol for clustered ad hoc networks, in: Distributed Computing Systems, 2007. ICDCS '07. 27th International Conference on, 2007, pp. 1–8.
[21] S.H. Wu, C.M. Chen, M.S. Chen, An asymmetric and asynchronous energy conservation protocol for vehicular networks, IEEE Trans. Mobile Comput. 9 (2010) 98–111.
[22] S.H. Wu, M.S. Chen, C.M. Chen, Fully adaptive power saving protocols for ad hoc networks using the Hyper Quorum System, in: Proceedings The 28th International Conference on Distributed Computing Systems, ICDCS 2008, 2008, pp. 785–792.
[23] W. Shan-Hung, C. Ming-Syan, C. Chung-Min, Optimally adaptive power-saving protocols for ad hoc networks using the Hyper Quorum System, Netw. IEEE/ACM Trans. 22 (2014) 1–15.
[24] S.H. Wu, C.M. Chen, M.S. Chen, AAA: asynchronous, adaptive, and asymmetric power management for mobile ad hoc networks, in: Proceedings - IEEE INFOCOM, 2009, pp. 2541–2545.
[25] W.-S. Luk, T.-T. Wong, Two new quorum based algorithms for distributed mutual exclusion, in: Proceedings International Conference on Distributed Computing Systems, 1997, pp. 100–106.
[26] S.D. Lang, L.J. Mao, A Torus quorum protocol for distributed mutual exclusion, in: Proceedings of the 10th Int'l Conf. on Parallel and Distributed Computing and Systems, New Orleans, USA, 1998, pp. 635–638.
[27] S. Lai, Duty-Cycled Wireless Sensor Networks: Wakeup Scheduling, Routing, and Broadcasting, Computer Engineering,Virginia Polytechnic Institute and State University, 2010.
[28] G. Lu, B. Krishnamachari, Minimum latency joint scheduling and routing in wireless sensor networks, Ad Hoc Netw. 5 (2007) 832–843.
[29] R. Bhandari, Optimal physical diversity algorithms and survivable networks, in: IEEE Symposium on Computers and Communications Proceedings, 1997, pp. 433–441.
[30] J.W. Suurballe, R.E. Tarjan, Quick method for finding shortest pairs of disjoint paths, Networks 14 (1984) 325–336.
[31] J.W. Suurballe, Disjoint paths in a network, Networks 4 (1974) 125–145.
[32] MICAz datasheet, 2011. ⟨www.xbow.com⟩.
[33] Full details on AIMMS and associated solvers. [Online]. Available: http://www.aimms.com.
[34] IBM ILOG CPLEX optimizer [Online]. Available: www.ibm.com/software/integration/optimization/cplex-optimizer/.
[35] The Network Simulator—ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/.