Journal of Network and Computer Applications 35 (2012) 156–163

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

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

Scheduling mobile collaborating workforce for multiple urgent events ☆

Yuqing Sun [a,*], Dickson K.W. Chiu [b], Bin Gong [a], Xiangxu Meng [a], Peng Zhang [a]

[a] School of Computer Science and Technology, Shandong University, China
[b] Dickson Computer Systems, Hong Kong

ARTICLE INFO

Article history:
Received 15 September 2010
Received in revised form
13 January 2011
Accepted 24 February 2011
Available online 5 March 2011

Keywords:
Scheduling
Mobile workforce
Constraint

ABSTRACT

Despite the advancement of wireless technologies that allows collaboration at different places, under emergencies, professionals are often still required to arrive at the scene to carry out critical tasks. Under many practical constraints, how to schedule mobile collaborating workforce for urgent event requirements becomes a challenging problem. In this paper, we study the optimal mobile workforce assignment problems for multiple events and propose an efficient algorithm to find an optimal workforce arrangement with respect to quick response under qualification and location constraints. A practical example is given to illustrate how our method works. We also study the exception case where there are not enough qualified users. We allow a user to take on multiple qualified tasks previously assigned to different users. But each person is restricted within one event location so as to reduce traffic transfer between different places for the quick response purpose. We analyze the computational complexity of the problem of finding an optimal assignment of mobile workforce under such restraints and solve it by means of integer linear programming.

1. Introduction

Although intelligent handheld devices enable professionals to perform tasks via a variety of wireless technologies like WLAN, Bluetooth or GSM, under emergency situations (Goel et al., 2010; Sun et al., 2009), some professionals are often still required to arrive at the scene to carry out critical tasks, such as reconstruction of a damaged network, mobile healthcare, or maintenance of airplane (Sun and Chiu, 2010). In such situations, on one hand, there are basic requirements on the qualification and quantity for professionals, i.e. only enough qualified users can together fulfill the required tasks. For example, two surgeons and a Cardiologist are required in an urgent operation. On the other hand, for quick response requirement, the selected professionals should arrive at the scene as soon as possible and location is also an important constraint for the scheduling.

However, from the viewpoint of authorization management, since emergency events may happen anywhere and anytime, administrators may not be able to plan well on the assignment of mobile professional workforce and grant them all necessary permissions for such unpredictable tasks. In particular, when multiple events with different requirements happen at the same time, there is a practical limitation on the number of available professionals with the required qualifications (Naveh et al., 2007). How to schedule the professional workforce becomes a challenging problem. If there are more than one qualified candidates for a task, the candidates' physical location and the event location should be taken into account in the assignment. Only such assignment solution is found, can the required tasks be carried out properly.

There are many literatures work on the problem of scheduling workforce from different points of view. Some discuss how to choose the criteria for arrangement and some aim to find employee shift arrangements to match a time-varying customer demand for service while keeping cost under control (Andersen and Petersen, 1993; Castillo et al., 2009). Some discuss the problem of task-users assignment in access control systems and take the location condition into consideration (Ray and Kumar, 2006; Damiani et al., 2007; Ardagna et al., 2006). However, the current works do not discuss how to find an optimal mobile workforce assignment with respect to quick response for unpredicted events.

In this paper, we tackle the problem of scheduling mobile workforce for multiple events under qualification and location constraints (*MWA*). This significantly extends our earlier work (Sun et al., 2009), which only attempts to fulfill a single event at a time. The concept of event requirement is proposed to represent a dynamic context, which is the basis of our consideration. We study the optimal workforce assignment problem MWA_{opt}, which finds

☆ Parts of this paper were published in preliminary form in Proceeding of the 14th International Conference on Computer Supported Cooperative Work in Design, April 2010.

* Corresponding author. Tel.: +86 531 88391810.
E-mail addresses: sun_yuqing@sdu.edu.cn (Y. Sun), dicksonchiu@ieee.org (D.K. Chiu), gb@sdu.edu.cn (B. Gong), mxx@sdu.edu.cn (X. Meng), algzhang@sdu.edu.cn (P. Zhang).

1084-8045/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2011.02.008

an optimal arrangement with respect to quick response while satisfying all events requirements. An algorithm is proposed to solve MWA_{opt} and a practical example is given to illustrate how it works. We also study the exception case where there are not enough qualified users. The solution allows a user taking on multiple tasks previously assigned to different users. To meet the quick response purpose, we make practical restraints on the task assignment. We restrict each qualified person within one event location so as to reduce the traffic transfer between different places. We analyze the computational complexity of the problem of finding an optimal assignment of mobile workforce and solve it by means of integer linear programming.

The reminder of this paper is organized as follows. In the next section, we compare the related works. Section 3 formally presents some concepts required in this paper. In Section 4, we study the MWA_{opt} problem and present an algorithm to solve it, followed by a case study in Section 5 to illustrate our solution. In Section 6, we discuss how to manage the exception case. At last we conclude in Section 7.

2. Related work

The problem of scheduling workforce has long exist in practice. Generally, it aims to find employee shift arrangements to match a time-varying customer demand for service while keeping cost under control and satisfying some applicable regulations (Andersen and Petersen, 1993). Some discuss how to extend the cost component of the objective function to account for the cost of poor service and the cost of waiting (Easton and Goodale, 2005). Castillo et al. (2009) propose a paradigm, where cost minimization and service level maximization are considered. However, most of these works discuss how to choose the scheduling criteria. They do not present an exact solution on how to find such assignments to meet different requirements.

Our work seems related with the problem of task-users assignment in access control systems. In conventional applications, users are often pre-assigned necessary privileges to execute a task. For example, permissions in a role based access control (RBAC) (Sandhu et al., 1996) system are associated with roles, and users are granted permissions through the assigned roles. However, they are not quite suitable for the dynamic qualification requirements under emergencies since one could not predicate what permissions are required to assign a user for an unknown task. Moreover, they do not take the location factor into account for the assignment. Recently, location is considered as an important issue of context and is introduced into authorization decision (Ray and Kumar, 2006; Damiani et al., 2007). The location-based access control technologies allow taking users physical location into account when determining their eligible tasks. For example, Ardagna et al. (2006) integrate location-based conditions to grant or deny access by checking the requester's location as well as credentials. However, they do not discuss how to schedule mobile workforce for multiple urgent events.

As for our previous work (Sun et al., 2009), we discuss the quick response problem with consideration of geospatial information, but focusing on scenarios with only one emergency event. In fact, there are many distinct characteristics with the scenario of multiple events, such as different event places, multiple teams requirements, etc. Therefore, scheduling workforce for multiple events is much more sophisticated and especially difficult under the practical limitation on both user number and user qualifications. Our work may also look like traditional task assignment problems. However, a distinct difference is that we consider the qualification relationship in our problem, which is not discussed in these works. The qualification requirements for task performers are practical in reality. Although the qualification is considered in authorization management in Sun et al. (in press), there are two apparent differences. One is that the geographical information was not considered in their work, which plays a very important role in scheduling mobile workforce. Another is that they only find a valid solution without discussion on how to find an optimal one with respect to quick response. Especially under emergencies with multiple events happening at different places, the requested qualification is unknown in advance and therefore the problem of scheduling mobile workforce becomes much more complex.

3. Basic terminologies

In this section, we present the formalization of the required basic terminologies, together with illustrative examples.

3.1. Location context and location-based predicate

Context is used to describe the circumstances and settings of users and events in this paper which includes user current location, event location, etc. In a scenario of mobile application, each user is associated with a location-aware mobile terminal, with which one can request information services provided by an application server. Users are mobile and often work at different places. Their geographical information can be acquired and mapped to a logical and device-independent position in a system with the help of widespread deployment of location services, such as GPS, trilateration, triangulation, hyperbolic, etc.

Previous literatures have well studied various location-based predicates for such services (Ardagna et al., 2006; Hong et al., 2007; Marsit et al., 2005). For example, *Dis*(*u*,*pos*) estimates the distance of a specific user *u* from a given position *pos*, *Deter*(*u*,*region*) determines whether a user *u* is in a specific area *region*. With consideration of more environment contexts such as traffic situation, the required time for a user to arrive at some place can be estimated by the predicate *Estimate*(*u*,*pos*) (Ng and Chiu, 2006; Marsit et al., 2005). In this paper, we would directly adopt these predicates in solving the problem of mobile workforce assignment without repeating the details of their implementation. Interested readers can refer to the above bibliography.

3.2. Conditions and satisfaction

In practice, every collaborative task may have qualification requirements for the performers. In this paper, we define such qualification requirements as *Condition* and formulate it as a logic expression consisting of terms. A term specifies one aspect of qualification requirements, which can be expressed in the form of a boolean predicate defined by an administrator or event requester. For example, IS(*Degree*, "PhD") verifies whether a user has a *PhD* degree; IS(*role*, "doctor") specifies the requirement on the roles a user is taking on. Also the location-based predicates like *Estimate*(*u*,*pos*) mentioned in the previous subsection can be used in a term. Based on such terms, we present the notion of condition on performer qualification.

Definition 1 (*Condition, Con*). A condition ϕ is defined as a logic expression consisting of terms and operators in $\{\neg, \vee, \wedge\}$. It is in the disjunctive normal form (DNF) and is defined as follows:

- A qualification term is a condition.
- If ϕ_1 and ϕ_2 are conditions, then $(\phi_1 \wedge \phi_2)$ and $(\phi_1 \vee \phi_2)$ are also conditions, where $\vee$ and $\wedge$ are logic disjunctive and conjunctive operations, respectively.

- If ϕ is a condition, then $(\neg\phi)$ is also a condition, where $\neg$ is logic negation operation.

Intuitively, a condition describes the qualification requirements for an independent user in an event and each element specifies an elective condition on the required performer. Having a condition, we need to ensure no conflict exists. Since a condition is of DNF form, say $\phi=\phi_1\vee\phi_2\vee\cdots\vee\phi_m$, a conflict may happen only in its sub-formulas $\phi_i, i\in[1..m]$, namely some contradiction sub-conditions. For example, if there is a sub-condition $\phi_i=\phi_{ij}\wedge\neg\phi_{ij}$, no user can be qualified for it. ϕ_i would be useless and discarded. On the contrary, if a condition is a tautology, every user is qualified for this term.

To find the qualified users for the task represented by a condition, we need to evaluate users' attributes with the condition. In the access control system of an organization, all entities and their relationships are called *System State*, denoted as SyST in this paper. Users represent collaborating professionals and are assigned responsibilities to perform certain job functions. A user u being qualified for any term ϕ_i is evaluated TRUE for ϕ under the current SyST and Context, denoted as $Sat_\phi(u)$. For example, there are two users u_1=*Alice*, who has worked as a *Surgeon* for 5 years, and u_2=*Bob*, who just begins his Gynecologist career. If $\phi=\mathsf{IS}(role,Gynecologist)\vee(\mathsf{IS}(role,Surgeon)\wedge\mathsf{LAG}(work,3\ years))$, Alice and Bob, respectively, satisfy a sub-condition of ϕ. Thus $Sat_\phi(u_1)$ and $Sat_\phi(u_2)$ hold. In the rest of the paper, we would abstract away the details on constraints description.

We define U_ϕ as the set of all qualified users for a given condition ϕ in a system state. A condition can be satisfied under given SyST and Context if and only if there is a user whose attributes are evaluated "TRUE" for this condition. In another words, a condition can be satisfied if and only if $U_\phi\neq\emptyset$. Since a qualification term actually denotes a set of users who have some common attributes or common job responsibilities, we present the following rules to calculate the user set U_ϕ of all qualified users for ϕ:

- Case ϕ is a term ϕ_i: $U_\phi=U_{\phi_i}$.
- Case ϕ is the conjunction of conditions $\phi_i\wedge\phi_j$: $U_\phi=U_{\phi_i}\cap U_{\phi_j}$, where $\cap$ is the intersection operation of sets.
- Case ϕ is the disjunction of conditions $\phi_i\vee\phi_j$: $U_\phi=U_{\phi_i}\cup U_{\phi_j}$, where $\cup$ is the union operation of two sets.
- Case ϕ is the negation of a condition ϕ_i: $U_\phi=U-U_{\phi_i}$, namely the complement set of U_{ϕ_i} in U.

3.3. Event requirement and valid workforce assignment

Under emergencies, multiple events may happen at different places and multiple separate groups of collaborating professionals are required. The concept of event requirement is introduced to describe an emergency situation.

Definition 2 (*Event Requirement, ER*). An event requirement is defined as a tuple $\varepsilon=(\Phi,pos)$, where $\Phi=\{\phi_1,\phi_2,\ldots,\phi_l\}$ is a limited set of conditions, l is an integer, each element $\phi_i, i\in[1..l]$ is a condition representing the requirements for a standalone person, and *pos* is the event site.

An ER actually specifies the number and qualification on the user team required in an event, as well as the event location context. Hereafterwards, we adopt the notions $\varepsilon.\Phi$, $\varepsilon.\Phi.\phi_i$ and $\varepsilon.pos$ to denote each element of ε, respectively. For the case there are multiple events happen at different places, the overall situation is represented as a set $\hat{E}=\{\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_k\}$, where $\varepsilon_i\in\mathsf{ER}, i\in[1..k]$. Let $|\varepsilon|$ denotes the number of required users in a given event requirement ε, i.e. $|\varepsilon|=|\varepsilon.\Phi|$. Thus the total number of required users in $\hat{E}$ is $\sum_{i=1}^{k}|\varepsilon_i|$. To meet such emergency requirements, a valid workforce assignment is required, which should satisfy the requirements under current system state and context.

Definition 3 (*Valid Workforce Assignment*). Given a system state SyST, a set of event requirements $\hat{E}$ and a same-sized set of user sets $\hat{U}$, namely $|\hat{U}|=|\hat{E}|$, $\hat{U}$ is called a valid workforce assignment for $\hat{E}$ if and only if there is a one-one mapping relationship between $\hat{U}$ and $\hat{E}$ satisfying $\forall U_i\in\hat{U}$, there is a unique corresponding $\varepsilon_i\in\hat{E}$ such that $|\varepsilon_i|=|U_i|$ and each user $u_{ij}\in U_i$ satisfies a corresponding condition $\varepsilon_i.\Phi.\phi_j$ under SyST, say $Sat_{\varepsilon_i.\Phi.\phi_j}(u_{ij})$. Hereafter, we adopt U_{ε_i} to denote the user set mapping to ε_i.

In the following sections, we would discuss a series of problems on how to find a valid workforce assignment for a given event requirements under current context and system state.

4. The optimal mobile workforce assignment problem

As mentioned, under emergencies, multiple events may happen at different places and multiple separate groups of users are required. An administrator need to verify whether there are adequate professionals and find a solution of coordinating them to fulfill the tasks. Furthermore, for the purpose of quick response, one may wish to find an *optimal* assignment such that the required team can arrive at scene as quickly as possible. In this section, we discuss how to find a valid and optimal assignment.

Definition 4 (MWA_{opt}). Given a system state SyST and a set of event requirements $\hat{E}=\{\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_k\}$, $\varepsilon_i=(\Phi_i,pos_i)\in\mathsf{ER}$, $i\in[1..k]$, the optimal workforce assignment problem (MWA_{opt} for short) finds a valid workforce assignment $\hat{U}$ such that

- The elements of $\hat{U}$ are pairwise disjoint set of users, namely $\forall i,j\in[1..|\hat{E}|], i\neq j, U_i,U_j\in\hat{U}, U_i\cap U_j=\emptyset$.
- The total time estimated for all users in $\hat{U}$ to arrived at the event positions is the shortest.

In the MWA_{opt} problem, we emphasize the *one–one mapping* relationship between event requirements and users in a valid workforce assignment. This is based on the consideration that a standalone user is required for each task represented by a condition (otherwise, multiple conditions can be combined into one). Solving this problem could help an administrator to verify whether the current system state is appropriate for the task requirements. In the following subsection, we present an algorithm to perform this check and find a solution to schedule the workforce according to the urgent requirements and context.

4.1. An efficient solution to MWA_{opt}

In this section, we present an efficient polynomial solution to MWA_{opt} by reducing it to the Minimum(Maximum) Weight Perfect Matchings problem (MPM for short). There are three parts in our algorithm. First, we construct a weighted bipartite graph based on a given system state, context and event requirements. Then, we solve the MPM problem for the constructed graph. Finally, we can translate the returned results from MPM back to the solution for MWA_{opt}. An optimal mobile workforce assignment could be found if there are enough qualified users for the event requirements. Otherwise, a predefined specific value would be in the resulted set of edges.

The reason that we reduce MWA_{opt} to an existing graph problem rather than designing a specific algorithm for it is based on the following observations. The optimal workforce assignment problem is actually a combinatorial optimization problem and we want to benefit from the existing results in solving similar problems.

Our problem can be naturally represented as the minimum weighted match in weighted bipartite graphs (Cook and Rohe, 1999). In a weighted bipartite graph $G=(U,V,E)$, the vertex sets U and V are of the same size, i.e. $|U|=|V|$, and each edge e in $E \subseteq U \times V$ is associated with a weight $w(e)$. A *matching* is a subset of the edges, no two of which are incident with a common vertex. A *matching* M is perfect if each vertex is incident with exactly one member of M. The Minimum Weight Perfect Matchings problem is to find an optimal matching that minimizes the total weights $\Sigma_{e \in M} w(e)$. Previous literatures have well studied this problem and presented many efficient polynomial algorithms (Fukuda and Matsui, 2006), such as the *Hopcroft–Karp* algorithm (Ahuja et al., 1993). In the MWA_{opt} problem, users, event requirements, and the qualification relationship between them can be represented as a bipartite graph. If we make some *tricks* on this graph, it would satisfy the background of the Minimum Weight Perfect Matchings problem. Thus we can benefit from the efficient results directly. We present the details of our algorithm as follows.

Construction of a bipartite graph. Given a system state SyST, a set of events requests $\hat{E}=\{\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_k\}$, where $\varepsilon_i \in$ ER, $i \in [1..k]$, and the Context including predicates $ElapseT(u_i,\varepsilon_j.pos)$, where $u_i \in U, j \in [1.k]$, there may exist more than one optimal assignment. But the total cost of each is the same. Finding one of them is enough. We construct a bipartite graph $G=\langle A,B,E \rangle$ according to the above given data, where A and B are vertex sets, E is the edge set. Fig. 1 illustrates the graph model employed for this problem, while the construction algorithm is given in Fig. 2

The Minimum Weight Perfect Matchings problem requires that the two node sets have the same size (otherwise there must exist a node that could not be matched), and there is a weighted edge between each pair of nodes. So, we firstly make a sanitizing check to make sure there are enough users to take on the tasks required by the event, which also ensure the following discussion based on the fact $|A| \geq |B|$. In our construction, set A denotes the users, while set B denotes the conditions. If a user u_i is competent for a condition ϕ_j in event ε_k, we create an edge between the nodes $a_i \in A$ and $b_j \in B$ and associate this edge with weight $Estimate(u_i,\varepsilon_k.pos)$. If $|A|=|B|$ holds and there is an edge between any pair of nodes in these two sets, the generated graph satisfies the prerequisites of the existing algorithms for the Minimum Weight Perfect Matchings problem. Otherwise, we need to generate enough fake edges or nodes, which do not influence the final result.

Considering the case $|A|=|B|$ but there are not enough edges, we may construct enough fake edges in phase III with weight *infinity* (∞), which guarantees that these *fake edges* would not be picked up during the computation of a perfect *matching* unless there is no feasible solution. *infinity* can be a large enough value, say, larger than the sum of all edge weights. When $|A|>|B|$, we create $|A|-|B|$ fake nodes in B, denoted as B', and create a fake edge (a_i,b'_j) for each pair of nodes $a_i \in A$ and $b'_j \in B'$ with weight 0. This guarantees that no matter which user is selected for the fake node, there is no influence to the final result.

Solving the MPM problem and translate results into the solution of MWA_{opt}. After constructing the bipartite graph $N=\langle A,B,E \rangle$, we could adopt a well studied efficient solution to generate the minimum perfect matching M for N. In M, there may exist three types of edges from the point of weights, "$w(e) \neq 0 \wedge w(e) < \infty$", "0" and "$\infty$". If for each condition node $b_j \in B$, there is an edge $e=(a_i,b_j) \in M$ satisfying $w(e) \neq 0 \wedge w(e) < \infty$, an optimal workforce assignment solution is found and user u_i is assigned the task represented by ϕ_j. The total elapse time for the selected users to be the site of assigned task is minimum.

Otherwise, an edge e under $w(e)=\infty$ existing in the result M indicates there does not exist a valid solution.

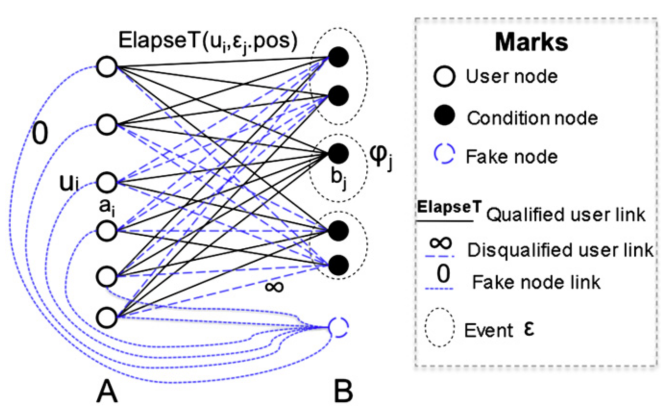

Fig. 1. A constructed bipartite graph based on a given system state, context and event requirements.

4.2. Analysis on the algorithm

Now, we prove that an optimal workforce assignment (solution to MWA_{opt}) is found under SyST and Context if and only if for the generated weighted bipartite, the Minimum Weight Perfect Matchings problem returns a perfect matching M such that the total weights $\Sigma_{e \in M} w(e)$ is minimum.

Proposition 1. *For a given system state* SyST, Context *and event requirements* $\hat{E}=\{\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_k\}$, *there is a solution to* MWA_{opt} *if and only if for the bipartite graph N constructed by the algorithm of* Fig. 2, *there is no* ∞ *edge in the final result of minimum weighted perfect matching M, namely each selected edge in M is either* $Estimate(u_i,\varepsilon_j.pos)$ *or* 0, *where* $u_i \in U$ *and* $\varepsilon_j \in \hat{E}$.

Phase I: Pre-process.
- If $|U| < |\hat{E}|$, say there is not enough users to take on the tasks required by the events, then return False.
- For each user $u_i \in U$ create a node a_i in set A

Phase II: Handling qualified users.
- For each event requirement ER $\varepsilon_k \in \hat{E}$
  - For each condition $\phi_j \in \varepsilon_k$
    - Create a node b_j in set B
    - Calculate the predicate U_{ϕ_j}
  - For each user $u_s \in U_{\phi_j}$
    - Calculate the approximate elapse time for user u_s to be event scene $\varepsilon_k.pos$: $Estimate\ (u_s, \varepsilon_k.pos)$
    - Create an edge $e=(a_s, b_j)$ between a_s and b_j with weight $w(e)=\ Estimate\ (u_s, \varepsilon_k.pos)$ in set E

Phase III: Addition of fake nodes and edges.
- For each pair of nodes a_i and b_j without an edge
  - Create a fake edge with weight infinity ∞ in set E
- If the size of set A is larger than the size of set B, say $|A| > |B|$
  - Create total $|A|-|B|$ fake nodes in B, denoted as B'
  - For each node $b'_j \in B'$, associate each edge (a_i, b'_j) with weight 0

Fig. 2. The construction algorithm: for given SyST, Context and a set of events requests $\hat{E}$, we construct a weighted bipartite graph $N=\langle A,B,E \rangle$.

First, let us assume that there is a workforce assignment $\hat{U}=\{U_1,U_2,\ldots,U_k\},U_i \subseteq U,i \in [1..k]$ for the event requirements $\hat{E}$ under SysT and Context. That is to say, every user u_j in U_i, $i \in [1..k],j \in [1..|U_i|]$, satisfies a condition ϕ_j in ε_i and is assigned the task represented by ϕ_j. For convenience and without loss of generality, we map u_{ij} to ϕ_{ij}. Now, we construct a perfect matching M according to the above workforce assignment in the following way.

For every task-user assignment u_{ij} to ϕ_{ij}, the edge $e=(a_{ij},b_{ij})$ is added to the matching M. According to the construction of the bipartite graph, there are three types of edge. A normal edge with weigh $Estimate(u_i,\varepsilon_j.pos)$ denotes user u_i being competent for a condition in event ε_j. A fake edge between a user and a condition means one is not competent for this task. Since its weight is infinity, it would never be selected in the perfect matching if there were other non-infinity edge joined to it. The third type is a fake edge with weight 0 between a user node and a fake condition node. For each user u_m not included in any U_i of $\hat{E}$, we randomly select a fake node b_n and add the edge (a_m,b_n) to M. Since such edge is of the third type, it does not influence the final matching weight no matter how many such edges are selected. Overall, the sum of edge weight in M is $\Sigma_{e \in M} w(e)=\Sigma_{e=(a_{ij},b_{ij})} Estimate(u_{ij},\varepsilon_i.pos)$. In general, we have proved that the matching M is perfect during our construction.

On the other hand, assume there is a perfect matching M for the constructed weighted bipartite graph N, where no edge weight is *infinity*. According to the construction of N, any *fake* edge between a user node and a condition node could not be selected since its weight is *infinity*. Thus, only two kinds of edges may exist in M, either linking a qualified user with an exist condition node, or linking a user with a fake condition node. Now we construct user sets $\hat{U}=\{U_1,U_2,\ldots,U_k\},U_i \subseteq U,i \in [1..k]$ according to the following rules: for each edge $e=(a_{ij},b_{ij}) \in M \wedge w(e) \neq 0$, we add user u_{ij} to set U_i with assigned task ϕ_{ij} in ε_i. Since M is the minimum perfect matching of the graph and edge weight denotes the time cost for users to be present at the event position, the generated set $\hat{U}$ is an optimal workforce assignment for $\hat{E}$ under SysT and Context.

Now we analyze the time complexity of our solution, which is polynomial. Our solution includes three parts: (i) reduce to the Minimum Weight Perfect Matchings problem, (ii) solve the constructed weighted bipartite graph and generate the perfect matching, and (iii) arrange the workforce according to the resulted matching. In the reduction part, the complexity is bounded by the size of user set and the number of unit terms of the given event requirements, denoted as $|U|$ and n_ϕ henceforward, respectively. Then we need to create $|U|$ user nodes, n_ϕ condition nodes and create an edge between each pair of nodes in two sets. Since $Estimate(u_i,\varepsilon_j.pos)$ can be adopted directly as we discussed before, the computational complexity of this part is bounded by $O(|U|*n_\phi)$.

In the second part of solving the Minimum Weight Perfect Matchings problem, there are many efficient methods (Cook and Rohe, 1999). For example, Gabow's algorithm is bounded by $O(|V|(|E|+|V|\log|V|))$ in its worst case (Gabow, 1990), where $|V|$ is the number of vertexes and $|E|$ is the size of edge set (bounded by $|u|*n_\phi$). Finally, we need to transform the resulted minimum-weight perfect matching M to the solution of MWA_{opt}, which is by the size of M, namely n_ϕ. From above analysis, we can see that although the time complexity depends on real system state and event requirements, our solution is still efficient.

5. An illustrative example

In this section, we would present a comprehensive example to illustrate what a system state may be and how to specify practical event requirements, as well as how to solve MWA_{opt} with our method.

Consider a mobile healthcare delivery application in a hospital. Individuals are called users from the system view and are assigned responsibilities to perform certain job functions. Each of them is given a location-aware mobile terminal, with which one can request information services provided by the application server. The organizational roles like nurse, doctor, patient, and so on, are associated with different functions and access permissions. Let U and R denote the set of users and organizational roles in the system, respectively. Role hierarchies $RH \subseteq R \times R$ are the partial orders on R and define inheritance relations among roles, written as $\succeq$. The expression $r_i \succeq r_j$ means that users who are members of r_i are also members of r_j, while all permissions assigned to r_j are inherited by r_i. The user-role assignments are defined as the set of relationships $UR \subseteq U \times R$. A user is assigned to one or more roles to control task execution either by assignment or by inheritance from role hierarchies. But the roles available to them depend on their current context, such as geographical position, duty time, etc. The details are given as follows.

System state. $\mathsf{SysT}=\langle U,R,RH,URA \rangle$, where user set $U=\{Alice, Bob, Carl, Dan, Ellen, Frank, Gary\}$. The organizational role set is $R=R_1 \vee R_2=\{Cardiologist, Dermatologist, Gynecologist, Surgeon\} \vee \{Resident, Physician, AC.Physician, C.Physician\}$, where *AC.Physician* and *C.Physician* denote the professional titles of associate chief physician and chief physician, respectively. Role hierarchies are $RH=\{C.Physician \succcurlyeq AC.Physician, AC.Physician \succcurlyeq Physician, Physician \succcurlyeq Resident\}$. The user-role assignments (URA) are listed in Table 1.

Event requirements. Here we consider three emergency events $\varepsilon_1,\varepsilon_2$ and ε_3 happened at different places and three teams of various medical specialists are required to arrive at the event sites for medical treatment. The first event ε_1 requires two persons, where one is both a Cardiologist and a Dermatologist and another is a Gynecologist or a Surgeon whose work experiences are more than 3 years. The requirement for event ε_2 is one person who can be either a "Gynecologist and AC.Physician" or a "Surgeon and a Physician", while the requirements for event ε_3 are two persons: one is both Cardiologist and Physician and another is a Gynecologist. These requirements are expressed as follows:

$\varepsilon_1=(\{\phi_{11},\phi_{12}\},pos_1)$, $\varepsilon_2=(\{\phi_{21}\},pos_2)$, $\varepsilon_3=(\{\phi_{31},\phi_{32}\},pos_3)$, where
$\phi_{11}=\mathsf{IS}(role,Cardiologist) \wedge \mathsf{IS}(role,Dermatologist)$,
$\phi_{12}=\mathsf{IS}(role,Gynecologist) \vee (\mathsf{IS}(role,Surgeon) \wedge \mathsf{LAG}(work, 3\ years))$,
$\phi_{21}=(\mathsf{IS}(role,Surgeon) \wedge \mathsf{IS}(role,Physician)) \vee (\mathsf{IS}(role, Gynecologist) \wedge \mathsf{IS}(role,AC.Physician))$,
$\phi_{31}=\mathsf{IS}(role,Cardiologist) \wedge \mathsf{IS}(role,Physician)$,
$\phi_{32}=\mathsf{IS}(role,Gynecologist)$.

Context. Context includes users' current positions and the event scenes, which can be acquired via mobile equipments. The predicate $ElapseT(u_i,\varepsilon_j.pos)$ for each user $u_i,i \in [1..k]$, to arrive at the event scene $\varepsilon_j.pos,j \in [1..k]$, then can be calculated. In practice, we only calculate each pair of events and their qualified users. The simulated values of *ElapseT* are labeled on the edges in Fig. 3.

Table 1
User-role assignments for our case study.

User	Assigned roles
Alice	*Cardiologist, Surgeon, Resident*
Bob	*Dermatologist, Gynecologist, AC.Physician*
Carl	*Gynecologist, Surgeon, Physician*
Dan	*Surgeon, Resident*
Ellen	*Cardiologist, Dermatologist, Physician*
Frank	*Cardiologist, Gynecologist, C.Physician*
Gary	*Gynecologist, Surgeon, Resident*

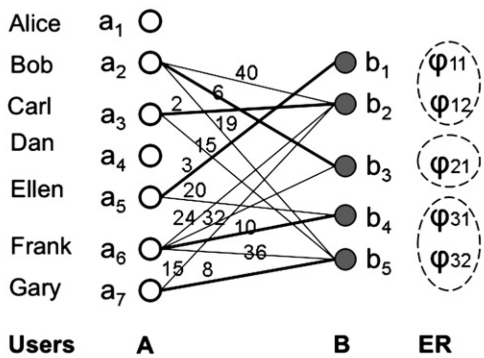

Fig. 3. The constructed original bipartite graph for users (nodes in set *A*), event requirement conditions (nodes in set *B*) and the qualification relationship between them (edges), where each edge weight is *ElapseT*(*u*,*pos*).

Table 2
The predicate U_ϕ on qualified users for each condition ϕ.

ϕ	U_ϕ
ϕ_{11}	*Ellen*
ϕ_{12}	*Bob, Carl, Frank, Gary*
ϕ_{21}	*Bob, Frank*
ϕ_{31}	*Ellen, Frank*
ϕ_{32}	*Bob, Carl, Frank, Gary*

Solving MWA_{opt} *under SyST and Context*. Under the current situation, we would verify whether a given system state satisfies the event requirements, and if yes, we would find the *optimal* workforce assignment for the purpose of quick response.

For each condition ϕ_{ij}, we calculate its qualified users via the predicate $U_{\phi_{ij}}$ and list the results in Table 2. Take the example of $\phi_{11} = \mathsf{IS}(role,Cardiologist) \wedge \mathsf{IS}(role,Dermatologist)$, $U_{\phi_{11}} = U_{\mathsf{IS}(role,Cardiologist)} \cap U_{\mathsf{IS}(role,Dermatologist)} = \{Alice,Ellen,Frank\} \cap \{Bob,Ellen\} = \{Ellen\}$.

Based on above system state SyST, Context, and events requirements, the original bipartite graph $N = \langle A,B,E \rangle$ is constructed as shown in Fig. 3, where each user corresponds to a node in set *A* and each condition corresponds to a node in set *B*. The weight of each edge between a condition node and a qualified user is set to *ElapseT*(*u*,*pos*). There is no fake edge and fake node in the original graph. For example, Alice and Dan are not qualified for any event requirement.

In the extended graph, we construct necessary fake nodes and fake edges for the Minimum Weight Perfect Matchings problem and set *infinity* as an enough large value, like 10,000 in this case (larger than the sum of all edge weights in the original graph). The final constructed bipartite graph is similar to Fig. 1, which is omitted here. The matching result for solving the MPM problem of above graph is $M = \{(a_5,b_1,3),(a_3,b_2,2),(a_2,b_3,6),(a_6,b_4,10),(a_7,b_5,8)\}$, as the bold lines shown in Fig. 3. We construct a solution to the MWA_{opt} problem and schedule the workforce as three user teams with assignments $\{(Ellen,\phi_{11}), (Carl,\phi_{12}), (Bob,\phi_{21}), (Frank,\phi_{31}), (Gary,\phi_{22})\}$ and $\tilde{U} = \{U_{\varepsilon_1},U_{\varepsilon_2},U_{\varepsilon_3}\}$, where

$$U_{\varepsilon_1} = \{Ellen,Carl\}$$
$$U_{\varepsilon_2} = \{Bob\}$$
$$U_{\varepsilon_3} = \{Frank,Gary\}.$$

Note that if an *infinity* edge exists in the result, there would be no valid workforce arrangement solution for the event requirements, which means, the answer to MWA_{opt} is NO. For example, if a condition is $\phi = C.Physician \wedge Dermatologist$, no user is qualified for it under the above system state.

6. Handling exceptions

In above discussion, MWA_{opt} returns an optimal workforce assignment for urgent events with respect to quick response. However, in many cases there may not exist enough qualified users for a valid workforce assignment to make every task assigned to a standalone person. In this section, we discuss how to handle such exceptions. A possible solution is to allow a user to take on multiple qualified tasks. Under such consideration, what we need to do is to find a valid user-task assignment such that every task is assigned to a qualified user. For each condition in event requirements, we check whether the qualified user set U_ϕ is not empty. This check is simple and efficient, which can be processed in linear time by the number of total conditions. The details are omitted here.

However, if there is no restraint on the tasks one is allowed to take on, a user qualified for multiple conditions may be assigned too many tasks in different positions. Thus, the geographical transfer would cause too much delay, which is inefficient for quick response purpose . A practical restraint is to restrain a user only perform multiple tasks in one place so as to reduce the transfer time between different places. The mobile workforce scheduling problem under such restraint is introduced as follows.

Definition 5 (MWA_{one}). Given a system state SyST, and a set of event requirements $\hat{E}$, the mobile workforce assignment problem under location restraint (MWA_{one}) determines whether there is an optimal workforce assignment $\hat{U}$ where every user is assigned the tasks only involved in one event.

First, we study the computational complexity of MWA_{one}, and then present an algorithm for the solution.

Theorem 2. *The computational complexity of* MWA_{one} *is* **NP**-hard.

The proof of this theorem is given in Appendix A. Since MWA_{one} is **NP**-hard, there is no polynomial solution for it. We would adopt the Integer Linear Programming method (ILP for short) (Schrijver, 1998) to solve the MWA_{one} problem. Integer Linear Programming is a technique used to solve the problem of maximizing or minimizing a linear objective function subject to linear equality and inequality constraints. Problems in many fields can be expressed as linear programming problems, and there are many efficient ILP solvers that we can use directly. For example, *LP_SOLVE* is a user-friendly linear integer programming solver (LP_solve, 2010).

Now, we show how to model the MWA_{one} problem with Integer Linear Programming variables and constraints. For each condition ϕ_j in each event ε_i, i.e. $\phi_j \in \varepsilon_i.\Phi$, we compute the qualified user set U_{ϕ_j}. For each user $u_k \in U_{\phi_j}$, we specify a variable x_{ijk} denoting user u_k is competent for the task ϕ_j in event ε_i. The total number of variables is determined by the satisfaction relationship between conditions and users, namely bounded by $O(n_\phi * |U|)$. x_{ijk} can be 0 or 1. x_{ijk} being set to 1 indicates that user u_k is in the resulting workforce assignment U_{ε_i} and is assigned the task represent by ϕ_j. Otherwise, user u_k is not selected for this task. For the convenience of the following computation, we introduce the notion $UCan_{\varepsilon_i}(u_k)$ to denote the set of all conditions that u_k is qualified for in event *i*, i.e. $UCan_{\varepsilon_i}(u_k) = \{\phi \mid \phi \in \varepsilon_i \wedge Sat_\phi(u_k)\}$.

Then for each user u_k and each variable x_{ijk}, $i \in [1..|\hat{E}|]$, we specify a constraint

$$x_{ijk} + x_{i'j'k} = 1, \quad i \neq i',\ j \in UCan_{\varepsilon_i}(u_k)\ ,j' \in UCan_{\varepsilon_{i'}}(u_k). \tag{1}$$

Such constraints can ensure that a user cannot be assigned the tasks in any two events. Although the number of total constraints is the exponential size of qualification relationship between users and conditions, in practice it is small. This is because the MWA_{one} problem considers the exception case of inadequate qualified users, there is less qualified users for each condition. Otherwise, if there are many candidates for each task, we can have an *optimal* workforce assignment when solving the MWA_{opt} problem.

Finally, we specify a target linear function

$$Minimize: \sum_{i,j,k} x_{ijk} * ElapseT(u_k, \varepsilon_i.pos). \tag{2}$$

Another alternative restraint is an upper bound on the total tasks a user is allowed to take on so as to balance the workload among all users. We denote this restraint as follows.

Definition 6 (Upr_t). An Upr_t restraint $\text{Upr}_t(u,k)$, where $u \in U$ is a user and k is an integer, requires that the upper bound of tasks u is allowed to take on is k.

A restraint $rs = \text{Upr}_t(u,k)$ is satisfied if and only if u is assigned no more than k tasks. It is easy to understand that to ensure the consistency of multiple Upr_t restraints, a user can be specified on only one Upr_t restraint. When $k = \infty$, there is no restriction on the number of tasks a user may take on. If there were no explicit Upr_t restraint on a user, performing only one task is allowed in any event.

Definition 7 (MWA_{Upr_t}). Given a system state SyST, a set of Upr_t restraints and a set of event requirements $\hat{E} = \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k\}$, where k is an integer, $\varepsilon_i = (\Phi_i, pos_i) \in \text{ER}$, $i \in [1..k]$, the mobile workforce assignment problem under Upr_t restraints (MWA_{Upr_t}) determines whether there is a valid workforce assignment $\hat{U}$ satisfying Upr_t restraints.

The problem MWA_{Upr_t} can be also solved by the Integer Linear Programming method. The additional constraints are specified as follows. If there is a restraint $rs = \text{Upr}_t(u_k, l_k)$, we specify a constraint $\sum_{i,j,k} x_{ijk} \le l_k$, otherwise $\sum_{i,j,k} x_{ijk} \le 1$. If a user is still restricted in one event, Eq. (1) would be remained; otherwise it would be discarded.

7. Conclusions

Qualification and location are important factors in scheduling mobile collaborative workforce for urgent requirements. In this paper, we study the optimal mobile workforce assignment (MWA_{opt}) problems for multiple events under these practical considerations, which finds an optimal workforce arrangement with respect to quick response. An efficient algorithm is proposed to solve MWA_{opt} and a practical example is given to illustrate how our method works. To manage the exception case where there are not enough qualified users for event requirements, we allow a user to take on multiple tasks previously assigned to different users but to restrict each qualified person within one event location so as to reduce traffic transfer. Under such restraints, the problem of finding a feasible assignment of mobile workforce is **NP**-hard. We solve it by means of the integer linear programming.

As continuing work, we are investigating to consider more sophisticated context information and other criteria into scheduling. A potential direction is to add a different emergent degree to each event and the optimal target becomes to minimize the overall cost. Another future research direction is to investigate the secure delegation problem, which also handles exceptions and alternatives.

Acknowledgments

Part of this work is supported by the National Science Foundation (61070017), the Science Foundation of Shandong Province Project (Y2008G28) and the Independent Innovation Foundation of Shandong University (2010JC010).

Appendix A. Proof of Theorem 2

The proof of this theorem can be concluded from Section VI in Sun et al. (in press) by setting a mutual exclusion constraint for every pair of conditions in different event requirements. However, the number of such constraints would be exponential with the size of total conditions. Thus, in this section we present a direct way of reducing the **NP**-complete SAT problem to MWA_{one}.

In SAT, we are given an expression τ in conjunctive normal form (CNF) and are asked whether there exists a truth assignment for variables appeared in τ such that τ is evaluated to true. Let $\tau = \tau_1 \wedge \cdots \wedge \tau_k$, where $\tau_i = l_{i1} \vee \cdots \vee l_{il}$ is a clause and l_{ij} is a literal (i.e. a variable or the negation of a variable). Without loss of generality, assume that no clause contains both ν and $\neg\nu$. Let $\{\nu_1, \ldots, \nu_n\}$ be the set of variables appeared in τ.

Since it is not necessary to construct every element of SySТ, Context and $\hat{E}$, like what a condition exactly being, we only construct the key elements. Let $U = \{u', u'', u_1, \ldots, u_n\}$ and $E = \{\varepsilon_0, \varepsilon_1\}$, where $\varepsilon_0.\Phi = \{\phi_{1,0}, \ldots, \phi_{n,0}\}$ and $\varepsilon_1.\Phi = \{\phi_{1,1}, \ldots, \phi_{n,1}\}$. Intuitively, $u_i (i \in [1,n])$ corresponds to variable ν_i in the SAT instance; $\phi_{i,0}$ and $\phi_{i,1}$ correspond to setting variable ν_i to false and true, respectively. This indicates that variable ν_i cannot be set to both false and true. We construct k elements $\psi_1, \ldots, \psi_k$, where ψ_i corresponds to clause τ_i in the SAT instance. If ν_i appears in τ_j, we add ψ_j to event ε_1, namely $\varepsilon_1.\Phi = \varepsilon_1.\Phi \cup \{\psi_j\}$. Otherwise, if $\neg\nu_i$ appears in τ_j, we add ψ_j to event ε_0, namely $\varepsilon_0.\Phi = \varepsilon_0.\Phi \cup \{\psi_j\}$. Let u' be qualified for every task $\phi_{i,0}$ and u'' be qualified for every task $\phi_{i,1}$, $i \in [1..n]$. Each $u_i (i \in [1..n])$ is only qualified for $\phi_{i,0}$ and $\phi_{i,1}$ but not any other task $\phi_{j,0}$ and $\phi_{j,1}$ when $j \ne i$. Let $u_i (i \in [1,n])$ be qualified for every ψ_j if and only if variable ν_i or its negation appears in the clause τ_j. For every pair of user $u_i \in U$ and event ε_j, set $ElapseT(u_i, \varepsilon_j.pos) = 0$.

Now, we prove that τ is satisfiable if and only if there exists a feasible solution to MWA_{one}. On the one hand, assume that T is a truth assignment that satisfies τ. We now construct a workforce assignment for MWA_{one}. For every $i \in [1,n]$, if ν_i is true, we assign $\phi_{i,1}$ to u_i and $\phi_{i,0}$ to u'; otherwise, if ν_i is false, we assign $\phi_{i,0}$ to u_i and $\phi_{i,1}$ to u''. Also, if $\phi_{i,1}$ (resp $\phi_{i,0}$) is assigned to u_i, then ψ_j is assigned to u_i if and only if τ_j contains ν_i (resp $\neg\nu_i$); that is to say, ψ_j is assigned to u_i if and only if setting ν_i to true (resp false) satisfies the clause τ_j. According to above construction, these two conditions must in one event, which ensure that no user is assigned the tasks in different events. Since every $\tau_j, (j \in [1..m])$ is satisfied by T, every task represented by $\psi_j, j \in [1..k]$ is assigned to at least one user. Also, every task $\phi_{i,0}, \phi_{i,1}$, $i \in [1..n]$ is assigned to one user. Each user takes the tasks only in one event. Therefore, the workforce assignment is valid.

On the other hand, assume that there exists a workforce assignment to MWA_{one}. Since $\phi_{i,0}$ and $\phi_{i,1}$ are in different events, one of them is assigned to u_i and the other is assigned to u' or u''. We construct a true assignment to τ by setting variable ν_i to false if and only if $\phi_{i,0}$ is assigned to u_i; otherwise, we set ν_i to true. Now, we prove that every clause in τ is satisfied by the truth assignment. Assume that ψ_i is assigned to u_j. Since u_j is qualified for ψ_i, according to our construction, either ν_j or $\neg\nu_j$ appears in τ_i. Without loss of generality, assume that ν_j appears in τ_i. In this case, according to our construction, ψ_i and $\phi_{j,0}$ are in different events. Hence, u_j must have been assigned to $\phi_{j,1}$, which indicates that v_j is set to true, and τ_i is satisfied. This indicates that every clause in τ is satisfied and thus τ is satisfied.

References

Ahuja RK, Magnanti TL, Orlin JB. Network flows: theory, algorithms and applications. Prentice-Hall; 1993.

Andersen P, Petersen NC. A procedure for ranking efficient units in data envelopment analysis. Management Sciences 1993;39(10):1261–4.

Ardagna CA, Cremonini M, Damiani E, di Vimercati SDC, Samarati P. Supporting location-based conditions in access control policies. In: Proceedings of ACM symposium on information, computer and communications security. New York, USA; 2006. p. 212–22.

Castillo I, Joro T, Li YY. Workforce scheduling with multiple objectives. European Journal of Operational Research 2009;196(1):162–70.

Cook W, Rohe A. Computing minimum-weight perfect matchings. INFORMS Journal on Computing Archive 1999;11(2):138–48.

Damiani ML, Bertino E, Catania B, Perlasca P. GEO-RBAC: a spatially aware RBAC. ACM Transaction on Information System Security 2007;10(1):1–42.

Easton FF, Goodale JC. Schedule recovery: unplanned absences in service operations. Decision Sciences 2005;36:459–88.

Fukuda K, Matsui T. Finding all minimum-cost perfect matchings in bipartite graphs. International Journal of Networks 2006;22(5):461–8.

Gabow H. Data structures for weighted matching and nearest common ancestors with linking. In: Proceedings of the first annual ACM-SIAM symposium on discrete algorithms; 1990. p. 434–43.

Goel A, Gruhn V, Richter T. Lecture notes in business information processings. IBM Journal of Research and Development 2010;43:81–91.

Hong D, Chiu D, Cheung S, Shen E. Ubiquitous enterprise service adaptations based on contextual user behavior. Information Systems Frontier 2007;9(4):36–44.

LP_solve. Linear programming solver, 〈http://lpsolve.sourceforge.net/5.5/〉; 2010.

Marsit N, Hameurlain A, Mammeri F. Query processing in mobile environments: a survey and open problems. In: Proceeding of the first international conference on distributed framework for multimedia applications; 2005. p. 150–7.

Naveh A, Richter Y, Altshuler Y, Gresh DL, Connors DP. Workforce optimization: identification and assignment of professional workers using constraint programming. IBM Journal of Research and Development 2007;51(3):263–79.

Ng CW, Chiu D. e-Government integration with web services and alerts: a case study on an emergency route advisory system in Hong Kong. In: 39th Hawaii international conference on system sciences (HICSS39). IEEE; 2006.

Ray I, Kumar M. Towards a location-based mandatory access control model. Computers and Security 2006;25(1):343–58.

Sandhu RS, Coyne EJ, Feinstein HL, Youman CE. Role-based access control models. IEEE Computer 1996;29(2):38–47.

Schrijver A. Theory of linear and integer programming. John Wiley; 1998.

Sun Y, Chiu DKW. Context-aware scheduling of workforce for multiple urgent events. In: Proceedings of 14th international conference on computer supported cooperative work in design (CSCWD); 2010. p. 629–34.

Sun Y, Farwick M, Chiu DK. Constraint-based authorization management for mobile collaboration services. In: Proceedings of 2009 IEEE congress on services. Los Angeles, USA; 2009. p. 22–9.

Sun Y, Wang Q, Li N, Bertino E, Atallah M. On the complexity of authorization in RBAC under qualification and security constraints. Transaction on Dependable and Secure Computing (TDSC); in press. doi:10.1109/TDSC.2010.55.