Proceedings of the 2010 14th International Conference on Computer Supported Cooperative Work in Design

Context-Aware Scheduling of Workforce for Multiple Urgent Events

Yuqing Sun
School of computer Science and Technology
Shandong University, China
sun_yuqing@sdu.edu.cn

Dickson K.W. CHIU
Senior Member, IEEE
Dickson Computer Systems, Hong Kong
dicksonchiu@ieee.org

***Abstract*—Workforce management is an important issue for human involved collaboration. When multiple urgent events simultaneously happen at different places with unexpected requirements, administrators should schedule workforce under the consideration of both system and user context, such as event requirements, user qualifications, etc. In this paper, we tackle this challenging problem of scheduling workforce for multiple urgent events. We study the Feasible Workforce Assignment Problem (WAP) that determines whether a system state satisfies the requirements of a given multiple-event scenario. Then, we present an efficient polynomial solution to solve it, with a case study to illustrate the effectiveness of our method.**

***Keywords* — context awareness; scheduling; workforce; RBAC; multiple events**

I. INTRODUCTION

Workforce management is an important issue for human involved collaboration. To ensure effective performance of collaborative tasks or be compliant with business regulations, generally there are some minimal qualification requirements for a task performer. A system administrator can therefore assign the access permissions to users according to their qualifications. In the widely adopted Role Based Access Control (RBAC)[1] models, user qualifications are represented as roles, an abstract description of behavior and collaborative relations with others in an organization. Users are granted the permissions to execute a task only according to their assigned roles. For example, releasing funds is executed as a series of operations on documents or databases. Only the role *accountant* or the role *head* in the finance department are assigned with these permissions, and thus users who take on these roles can release funds.

Currently, with more and more mobile devices are being adopted in pervasive solutions for improving traditional collaboration methods, users can execute a task at different places via handheld devices, such as Personal Digital Assistants (PDA) or smart-phones. They are capable of communicating with similar devices and connecting to application servers with a variety of technologies, such as Wireless Local Area Network (WLAN), Bluetooth, or Global System for Mobile communications (GSM). In addition, these devices are capable of running reasonably complex applications, such as accepting task allocation, as well as perceiving context from external sources. These mobile applications are especially useful under emergency situations, such as disasters or terrorist attack, that require persons to arrive at the sites as quick as possible to carry out critical tasks. Another typical example is in the context of healthcare, where some solutions have been deployed to constitute a highly dynamic work environment, with the adoption of mobile devices for improving healthcare services. In the case that several serious patients outside the hospital requiring treatment, the hospital should schedule several teams of medical specialists to arrive at the scenes.

Under this scenario, with multiple emergency events happening simultaneously at different places, the requested tasks are unknown in advance and the workforce's qualification requirements may vary. Furthermore, there is a practical limitation on the number of workforce in any organization and not every member is qualified to perform every task. Scheduling workforce under such emergency situations becomes especially complex. A system administrator should take both user and event context into account for quick responses. This problem is also meaningful to evaluate whether the workforce in an organization and the access control configuration are ready for carrying out such critical tasks.

There are plenty of research works on context-aware access control systems [2], [3], [4], [5], and some of them take location based information into account to support dynamic collaboration [6], [7], [8]. For example, the spatially aware access control model GEO-RBAC [9] uses standard geospatial information to gain semantic knowledge of geospatial features and to map between coordinates and the logic location of users and objects. However, the existing works mainly focus on determining whether a specific subject (user) has access privileges to some objects when he/she is associated with a place, which do not tackle the problem on how to coordinate personnel for multiple urgent events.

Our contributions in this paper are as follows:

- We formalize various event requirements, such as user qualification and context, in the problem of context-aware workforce management, which captures the intuition discussed above.
- We formulate and study the Feasible Workforce Assignment Problem (WAP) that determines whether a system state satisfies the requirements of a given multiple-event scenario.
- We propose an efficient polynomial algorithm to solve

978-1-4244-6763-1/10/$26.00 ©2010 IEEE

WAP by reducing our problem to the MAXIMUM FLOW problem in order to benefit from the fruitful research results.
- We present a practical example to illustrate how our method works. The example is in a hospital scenario with complex system states and multiple events requirements.

The reminder of this paper is organized as follows. In the next section, we formally define event requirements and the WAP problem. Then we present a solution on solving WAP in Section III, followed by a case study in Section IV. Section V compares related work, before we conclude with the discussion on some open problems in Section VI.

II. THE WORKFORCE ARRANGEMENT PROBLEM

The Workforce Arrangement Problem (WAP) discussed in this paper is based on the Role Based Access Control Model (RBAC), since it is considered as the most appropriate paradigm for access control under complex scenarios [13]. In RBAC, role is an abstract description of behavior and collaborative relation with others in an organization. Permission is an access authorization to object, which is assigned to role instead of to individual user so as to simplify security administration. In many applications, a task is formulated as a series of permissions, such as releasing funds, is executed as a series of operations on documents or database.

The motivation of role hierarchies is to efficiently manage common permissions by defining multiple reusable subordinate roles in formulating other roles. Let R denote the set of organizational roles in a given system. Role hierarchies $RH \subseteq R \times R$ are the partial orders on a role set 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 tasks assigned to r_j are inherited by r_i.

Users are the subjects to be assigned responsibilities to perform certain job functions. In a system, a user is assigned to one or more roles to control user task executions. From the organizational view, there are minimal qualification requirements for role players in order to ensure effective performance or regulation compliance. Let U denote the set of users in a given system. The User-Role Assignments $UR \subseteq U \times R$ are implemented by evaluating a user's attributes against a role's qualification conditions. User u is assigned role r when his/her attributes satisfy the role qualification, denoted as $(u, r) \in UR$. For example, a doctor is assigned to the role $physician$ if and only he/she satisfies the qualification $certification =$ "PhD" $\wedge\, workinglife \geq$ "$5years$". In practice, a user u is allowed to take on a role r either by assignment or by inheritance from role hierarchies. We define the following two predicates $URole(u : U) \rightarrow 2^R$ and $RUser(r : R) \rightarrow 2^U$ to derive the roles that user u is allowed to play and the user set who are playing this role, respectively. Formally,

$URole(u) = \{r | r \in R \wedge ((u, r) \in UR \vee ((u, r') \in UR \wedge r \succeq r'))\}$

$RUser(r) = \{u | u \in U \wedge ((u, r) \in UR \vee ((u, r') \in UR \wedge r' \succeq r))\}$

We introduce the concept of $SystemState$ to denote all above entities and relationships between them in the access control system of an organization.

Definition 1 (System State)**.** A *System State* is given as a tuple $\langle U, R, UR, RH \rangle$, where U is a set of users, R is a set of roles, $RH \subseteq R \times R$ defines role hierarchies, and UR is the set of user-role assignments that any $(u, r) \in UR$, where $u \in U$ and $r \in R$, represents u is qualified for r.

A. Motivation Example

Consider a mobile healthcare delivery application in a hospital. Individuals are given location-aware mobile terminals, with which they 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, while individuals are assigned to these roles. An individual may take on several roles, but it does not mean that these roles are effective for them all the time. The roles available to them also depends on their current context. For example, only a doctor as a cardiologist on duty has the permissions to use the medical instruments and to access related patient records. Such mechanism further helps protect patient privacy and keep system security.

Suppose we are given the system state as Fig. 1. The organizational roles defined in this organization are $R = \{Cardiologist, Dermatologist, Gynecologist, Surgeon\}$, while the user set is $U = \{Alice, Bob, Carl, Dan, Ellen, Frank, Gary\}$ and each user is associated with several roles. A line segment connecting a user (e.g., Alice) to a role (e.g., Cardiologist) indicates that the user is assigned to the role or inherited the role via the role hierarchies. As discussed above, all the roles that user u is allowed to take on can be achieved by the predicate $URole(u)$. For example, the roles assigned to Bob are $Dermatologist$ and $Gynecologist$.

Suppose three emergency events concurrently happen in different places and three teams of various medical specialists are required to arrive at the sites for medical treatment. The event requirements are respectively expressed as:

- $Event_1$: one Cardiologist and one (Gynecologist and Surgeon);
- $Event_2$: two Cardiologists; and
- $Event_3$: one (Gynecologist and Surgeon) or one (Gynecologist and Dermatologist).

After receiving these requests, the organization need to check the system state whether there are adequate workforce to form three separate teams and each team consists of the required number of workforce whose qualifications satisfy the task requirements. That is to say, one team consists of two doctors where one is a Cardiologist and another is both a Gynecologist and a Surgeon; the second team is of two Cardiologists; and one doctor is required for the third team who is either both a Gynecologist and a Surgeon, or both a Gynecologist and a Dermatologist. In the following subsection,

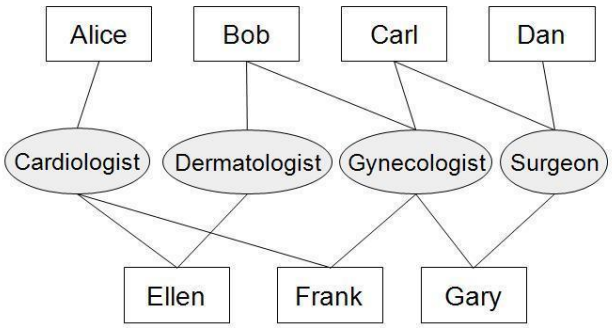

Fig. 1. User-role assignment of our illustrative example

we would discuss how to perform this check and schedule workforce for such urgent requirements.

B. Formal Definition of Event Requirements and WAP

In this subsection, we present the formal definition of event requirements and the Feasible Workforce Assignment Problem (WAP). Since here we focus on how to arrange the workforce for handling the emergency events, the event requirements are defined as the role qualifications for each required performer. But we need to state here, our method is not restrict to RBAC systems. A role in the expression is just a named set of users that can be replaced by users attributes or groups, which can then be applied to many other organizational models.

Definition 2. *(Person Requirement,* PR*) A person requirement ϕ is a logic term on role set, generally in the disjunctive normal form (*DNF*), that is defined as follows:*

- *A* term *is a role name(called unit term).*
- *If ϕ_1 and ϕ_2 are terms, then $(\phi_1 \wedge \phi_2)$ and $(\phi_1 \vee \phi_2)$ are also terms, where $\vee$ and $\wedge$ are logic disjunctive and conjunctive operations, respectively.*

Based on the previous predicate $RUser(r)$, we introduce the predicate $UserSet(\phi : \mathsf{PR}) : 2^U$ (U_ϕ for short) to calculate the users who satisfy the given ϕ. Formally, U_ϕ can be calculated as follows:

- Case ϕ is a unit term (i.e., a single role r): $U_\phi = RUser(r)$.
- Case ϕ is the conjunction of terms $\phi_i \wedge \phi_j$: $U_\phi = U_{\phi_i} \cap U_{\phi_j}$, namely the union set of $UserSet_{\mathsf{PR}}(\phi_i)$ and $UserSet_{\mathsf{PR}}(\phi_j)$.
- Case ϕ is the disjunction of terms $\phi_i \vee \phi_j$: $U_\phi = U_{\phi_i} \cup U_{\phi_j}$, namely the intersection of $UserSet_{\mathsf{PR}}(\phi_i)$ and $UserSet_{\mathsf{PR}}(\phi_j)$.

Definition 3. *(Satisfaction of a* PR*). Given a system state $\langle U, R, UR, RH\rangle$ and a* PR *ϕ, we say that ϕ can be satisfied if and only if the user set U_ϕ is not empty.*

In the above DNF PR form, say $\widetilde{\phi} = \phi_1 \vee \phi_2 \vee \cdots \vee \phi_m$, each element $\phi_i, i \in [1..m]$ denotes an elective condition on the required performer. Semantically, $\widetilde{\phi}$ can be satisfied if and only if there exist one or more persons such that the assigned roles of each person independently satisfy one of the disjunctive terms ϕ_i in $\widetilde{\phi}$. Obviously, there is no conflict in a PR term, since only conjunctive and disjunctive operations are used. From above discussion, we can see that negation operation is not necessary in this definition. Intuitively, we only need to list our wanted roles in a term.

Definition 4. *(Event Requirement,* ER*) An event requirement is a set of* PR *terms $\Phi = \{\widetilde{\phi_1}, \widetilde{\phi_2}, \cdots, \widetilde{\phi_k}\}$, each element $\widetilde{\phi_i}, i \in [1..k]$ is a* DNF PR *term representing the requirements of a standalone person.*

Intuitively, ER specifies the number and qualification of required user teams. An event requirement $\Phi = \{\widetilde{\phi_1}, \widetilde{\phi_2}, \cdots, \widetilde{\phi_k}\}$ can be satisfied if and only if there is a user set with size k, say $U_\Phi = \{u_1, u_2, \cdots, u_k\} \subseteq U$, such that each user u_i satisfies the qualification requirements in the corresponding term $\widetilde{\phi_i}$ in Φ. Considering the example of last section, the workforce requirements of the three events are respectively expressed as $\Phi_1 = \{\widetilde{\phi_1}, \widetilde{\phi_2}\} = \{Cardiologist, (Gynecologist \wedge Surgeon)\}$, $\Phi_2 = \{\widetilde{\phi_1}, \widetilde{\phi_2}\} = \{Cardiologist, Cardiologist\}$, and $\Phi_3 = \{\widetilde{\phi_1}\} = \{(Gynecologist \wedge Surgeon) \vee (Gynecologist \wedge Dermatologist)\}$. Based on ER, we present the formal definition of WAP Problem.

Definition 5. *(WAP). Given a system state $\langle U, R, UR, RH\rangle$ and a set of event requirements $\widetilde{\Phi} = \{\Phi_1, \Phi_2 \cdots, \Phi_m\}$, the Feasible Workforce Assignment Problem (WAP) determines whether there is a set $\widetilde{U} = \{U_{\Phi_1}, U_{\Phi_2}, \cdots, U_{\Phi_m}\}, U_{\Phi_i} \subseteq U, i \in [1..m]$, such that each user set U_{Φ_i} satisfies the corresponding* ER *Φ_i in $\widetilde{\Phi}$ under $\langle U, R, UR, RH\rangle$.*

Solving WAP could find whether there are enough qualified users to fulfill the tasks in these events and who they are for each event.

III. Solution on WAP

In this section, we would present a polynomial solution to WAP and analysis the computational complexity. By observing that our problem can be naturally represented as the Maximum Flow problem, we propose an algorithm to reduce our problem WAP to the Maximum Flow problem, and thus we can take the advantage of the fruitful research results in solving it. The Maximum Flow problem is to find a feasible flow through a single-source, single-sink flow network that is maximum. A flow is feasible if it does not exceed the capability on each edge of the flow network, and the total amount of incoming flows is equivalent to the total amount of outgoing flows on every node of the network, except the source and sink. The incoming flow of the source and outgoing of the sink is zero, while the outgoing flow of the source and incoming flow of the sink are equivalent to the flow. Maximum Flow is well-solved and many polynomial time algorithms have been worked out for it, such as the Ford-Fulkerson algorithm [10].

1.If $|U| < \Sigma_{i=1}^{m}(|\Phi_i|)$ then return False.
2.Create the source node s and and the sink node t in V.
3.For each user $u_i \in U$ create a node a_i and an edge $e = (s, a_i)$ between s and a_i
4.For each role $r_j \in R$ calculate the predicate $RUser(r_j)$
5.For each ER $\Phi_i \in \widetilde{\Phi}$
  6.For each PR $\widetilde{\phi}_j \in \Phi_i$
    7.create a node p_j and an edge $e = (p_j, t)$ between p_j and t
    8.calculate $U_{\widetilde{\phi}_j}$
  9.For each user $u_s \in U_{\widetilde{\phi}_j}$ create an edge $e = (a_s, p_j)$ between a_s and p_j
10.Set the capacity $w_e = 1$ of each edge $e = (v_i, v_j)$, where $v_i, v_j \in V$ are the two nodes of this edge.

Fig. 2. $Reduction_{WAP}$: for a given configuration $\langle U, R, UR, RH\rangle$ and events requirements $\widetilde{\Phi} = \{\Phi_1, \Phi_2 \cdots, \Phi_m\}$ in WAP, construct a flow network $N =< V, E, W >$.

A. The Algorithm

The reduction algorithm $Reduction_{WAP}$ is shown in Figure 2. By the given system state $\langle U, R, UR, RH\rangle$ and event requirements $\widetilde{\Phi} = \{\Phi_1, \Phi_2 \cdots, \Phi_m\}$, we would construct a flow network $N =< V, E, W >$, where V and E are the node set and edge set respectively, W is the integer function on each edge denoting its capacity. First, we make a sanitizing check to make sure that enough users are available for the events requirements. After establishing the source and sink nodes in step 2, we handle the user set in step 3. Each user is mapped onto a *user node* (i.e., a_i) in V.

Then we create a node (called PR $node$, i.e., p_j) for each PR $\widetilde{\phi}_j$ of every ER, in order to ensure a user would be assigned to this node. Afterward, for each ϕ_{jk} in PR $\widetilde{\phi}_j$, we check which users are qualified for this term and make links between node p_j with these users. This means a user satisfying any term ϕ_{jk} of $\widetilde{\phi}_j$ would be a candidate for PR. Now, we need to prove that we can get the correct answer to WAP by the algorithm of MAXIMUM FLOW under the constructed flow network N in $Reduction_{WAP}$.

Lemma 1. *For a given system state $\langle U, R, UR, RH\rangle$ and event requirements $\widetilde{\Phi} = \{\Phi_1, \Phi_2 \cdots, \Phi_m\}$, there is a solution to WAP if and only if for the constructed flow network N by the algorithm in Figure 2, there is a flow f from s to t such that $f = \Sigma_{(p_i,t)\in E} w_i$, where w_i is the capacity of the edge between p_i and t.*

Proof: First, let us assume that there is a task assignment $\widetilde{U} = \{U_1, U_2, \cdots, U_k\}, U_i \subseteq U, i \in [1..k]$ for the event requirements $\widetilde{\Phi}$ under $\langle U, R, UR, RH\rangle$. That is to say every user in U_i satisfies a term in Φ_i. Without lose of generality, we can assume that user $u_{ij} \in U_i$ maps to $\widetilde{\phi}_{ij}$ in Φ_i. Now, we construct a flow f according to the above workforce arrangement in the following way.

For every edge e between s and a_{ij} (mapping to above u_{ij}), there is a flow $f_e = 1$. Totally, the outgoing flows from s is $f' = \Sigma_{i=1}^{k}|U_i|$. For each pair a_{ij} and p_{ij}(mapping to the above $\widetilde{\phi}_{ij}$), there is a flow $f_e = 1$ on the edge $e = (a_{ij}, p_{ij})$. Finally, for every edge e between p_{ij} and t, there is a flow $f_e = 1$. Totally, the incoming flows for t is $f' = \Sigma_{e=(p_{ij},t)} f_e$, which is equivalent to f. According to the construction of the flow network, the flow on each edge is within the capacity. In general, we have proved that the flow f is a valid flow during our construction. Further, it is easy to observe from the last step of the construction that the flow is maximum.

On the other hand, assume that there is a flow f from s to t such that $f = \Sigma_{e=(p_i,t)} w_e$, where w_e is the capacity of the edge between p_i and t. We construct user sets $\widetilde{U} = \{U_1, U_2, \cdots, U_k\}, U_i \subseteq U, i \in [1..k]$ such that user $u_{ij} \in U_i$ maps to $\widetilde{\phi}_{ij}$ in Φ_i. Since $f = \Sigma_{e=(p_i,t)} w_e$, it must be the case that every edge between p_i and t is fully loaded. According to the construction of the flow network, every capacity of edge is 1 and thus every person requirement $\widetilde{\phi}_{ij}$ is assigned to exactly one user in U. Also, since the capability of the edge between s and a_j is 1 and the number of users is equal to or larger than the number of total PRs, which guarantees that the total amount of outgoing flow of s is no more than $|U|$. Finally, according to the construction of the flow network, there must be an edge from a_i to p_k with flow 1, where p_k maps to a unit term in PR $\widetilde{\phi}_k$. Suppose $\widetilde{\phi}_k \in \Phi_t$, then u_i is in the set U_t, which indicates that user u_i is assigned to the required task $\widetilde{\phi}_k$ in the event Φ_t. In general, $\widetilde{U}$ is a valid assignment for $\widetilde{\Phi}$ under $\langle U, R, UR, RH\rangle$. ■

B. Analysis on Computational Complexity

In this subsection, we analyze the time complexity of our solution, which is polynomial. Our solution includes two parts: (i) reduction to the Maximum Flow problem, and (ii) solving the constructed flow network. In the reduction part, the complexity of the first step is bounded by the number of unit terms of the given event requirements, denoted as n_Φ henceforward. The loop of step 3 takes the complexity of $O(|U|)$, where $|U|$ denotes the size of user set. Calculating $RUser(r)$ for all roles in next loop is related with both the size of assignments in UR and role hierarchies in RH, namely $O(|UR| * |RH|)$. If the role hierarchies are efficiently organized as an ordinary tree, the average complexity can be reduce to $O(|UR| * log^{|RH|})$. The next nested loops from step 5 to step 9 is the main part of our reduction. Actually the complexity of step 6 and 7 is bounded by n_Φ. Following the method given in section II-B, calculating all $U_{\widetilde{\Phi}_j}$ in step 8 is bounded by n_Φ also. However, the number of users in each set $U_{\widetilde{\Phi}_j}$ vary with real user-role assignments and role

TABLE I
USER-ROLE ASSIGNMENTS FOR OUR CASE STUDY

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

hierarchies of the given system state, but can never exceed the number of total users $|U|$. So, the complexity of nested loops is $O(n_\Phi * |U|)$. The overall complexity of the reduction is $O(|UR| * log^{|RH|} + n_\Phi * |U|)$.

In the second part of our solution, there are many efficient methods for solving the MAXIMUM FLOW problem. For example, Ford-Fulkerson's algorithm is in $O(|E|max|f|)$, where $|E|$ is the size of edge set and $|f|$ is the size of the resulted flow [10]. In our problem, if there is a feasible solution to WAP, the maximum flow must be equal to the number of total required persons, which is obviously equal or less than 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.

IV. CASE STUDY

In this section, we would present a comprehensive example, which is an extension of the example given in section II-A, to illustrate how to specify practical event requirements and how to solve WAP with our method.

a) System State: Suppose in a hospital scenario, the users, organizational roles, and role hierarchies have been defined, as well as the user-role assignments. Each set is given as follows: $U = \{Alice, Bob, Carl, Dan, Ellen, Frank, Gary\}$. $R = R_1 \vee R_2 = \{Cardiologist,\ Dermatologist,\ Gynecologist,\ Surgeon\} \vee \{Resident,\ Physician,\ AC.Physician,\ C.Physician\}$, where $AC.Physician$ and $C.Physician$ denote the professional titles of associate chief physician and chief physician, respectively. Role hierarchies are $RH = \{C.Physician \succeq AC.Physician, AC.Physician \succeq Physician, Physician \succeq Resident\}$. The user-role assignments are listed in Table I.

b) $RUser(r)$: The predicate $RUser(r)$ for each role is calculated and listed in Table II, which includes both assigned roles and inherited roles. This calculation can be performed only once for a given system state unless it is updated.

c) Event Requirements: Suppose there are three events happened at different places, namely $\widetilde{\Phi} = \{\Phi_1, \Phi_2, \Phi_3\}$. The requirements on workforce of each event are given as follows: $\Phi_1 = \{\phi_{11}, \phi_{12}\}$, $\Phi_2 = \{\phi_{21}\}$, and $\Phi_3 = \{\phi_{31}, \phi_{32}\}$, where

$\phi_{11} = Cardiologist \wedge Dermatologist$,

$\phi_{12} = Gynecologist \vee (Surgeon \wedge Physician)$,

TABLE II
THE PREDICATE $RUser()$ ON ROLES FOR OUR CASE STUDY

Role	$RUser(r)$
$Cardiologist$	$Alice, Ellen, Frank$
$Dermatologist$	$Bob, Ellen$
$Gynecologist$	$Bob, Carl, Frank, Gary$
$Surgeon$	$Alice, Carl, Dan, Gary$
$Resident$	$Alice, Bob, Carl, Dan$ $Ellen, Frank, Gary$
$Physician$	$Bob, Carl, Ellen, Frank$
$AC.Physician$	$Bob, Frank$
$C.Physician$	$Frank$

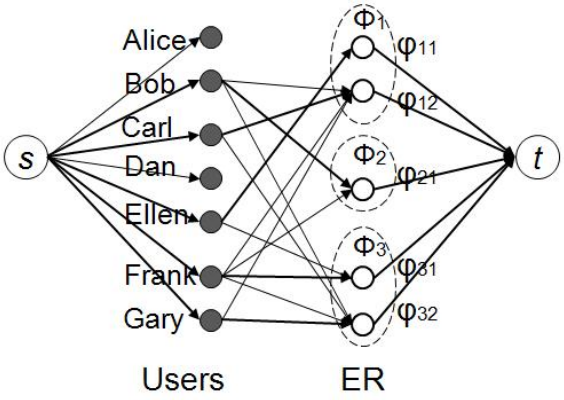

Fig. 3. The constructed flow network for our case study.

$\phi_{21} = (Gynecologist \wedge AC.Physician) \vee (Surgeon \wedge Physician)$,

$\phi_{31} = Cardiologist \wedge Physician$, and

$\phi_{32} = Gynecologist$.

d) Construction of the Flow Network: Based on above system state and events requirements, the flow network $N =< V, E, W >$ is constructed as Fig. 3. The weight on each edge is 1. We can see that users Alice and Dan are not qualified for any events requirements of PR.

The result of solving the MAXIMUM FLOW problem for this flow network is $|f|$=5, where the edges with non-zero flow are listed as follows and shown as the bold lines in Fig. 3:

$f_{(s,Ellen)} = 1$, $f_{(Ellen,\phi_{11})} = 1$, $f_{(\phi_{11},t)} = 1$,
$f_{(s,Carl)} = 1$, $f_{(Carl,\phi_{12})} = 1$, $f_{(\phi_{12},t)} = 1$,
$f_{(s,Bob)} = 1$, $f_{(Bob,\phi_{21})} = 1$, $f_{(\phi_{21},t)} = 1$,
$f_{(s,Frank)} = 1$, $f_{(Frank,\phi_{31})} = 1$, $f_{(\phi_{31},t)} = 1$,
$f_{(s,Gary)} = 1$, $f_{(Gary,\phi_{32})} = 1$, and $f_{(\phi_{32},t)} = 1$.

e) A Feasible Scheduling Result: Since the maximum flow is 5 and is qual to the number of total required persons in the multiple-event scenario, we have the answer to the WAP problem under the given state and the event requirements. We can schedule the workforce as three user teams. Each team U_{Φ_i} is for the corresponding event.

$\widetilde{U} = \{U_{\Phi_1}, U_{\Phi_2}, U_{\Phi_3}\}, where$

- $U_{\Phi_1} = \{Ellen, Carl\}$

- $U_{\Phi_2} = \{Bob\}$
- $U_{\Phi_3} = \{Frank, Gary\}$

There may exist more than one feasible solutions for the same case, where our solution only presents one of them. In case that the resulted maximum flow is less than the number of total required persons, the answer to WAP would be NO, which means there are unenough qualified users for the event requirements. For example, if a PR is $\phi = C.Physician \wedge Dermatologist$, no user is qualified for it under above system state.

V. Related Work

There exist a wealth of literatures on context-aware models and systems, such as the Task-Based Authorization Control model (TBAC) [2], the flexible security policy for active cooperation [11], the Activity-based Access Control model [3], and the Task-role-based Control Model (T-RBAC) [4], etc. They support dynamic task allocation for workflow or collaboration under different contexts by granting, tracking, and revoking permissions associated with process execution. However, from a conceptual standpoint, our considerations and method are significantly different and comprehensive. Those works mostly deal with predefined workflows, while we focus on the workforce management for multiple urgent events, where the required performer qualifications are unknown in advance.

Recently, location is considered as an important issue of context and is introduced into access control decision[7], [6], also called Location-based Access Control (LBAC). LBAC technologies allows taking users physical location into account when determining their eligible tasks. Considering the Role Based Access Control model (RBAC) [1] being widely adopted in a wide range of applications, some literatures try to integrate the location based information into RBAC to satisfy certain geographic requirements [8], such as GEO-RBAC [9]. However, these works mainly consider integrating the location based information into the access control decision, such as the specification and enforcement of location based conditions. They are not suitable for the scheduling of mobile workforce for multiple urgent events.

As for our previous work [5], we discuss the quick response problem with consideration of geospatial information, but focusing on scenarios with just one emergency event. In fact, there are many distinct characteristics with the scenario of multiple events, such as different event requirements, multiple team 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.

VI. Conclusions

Workforce management is an important issue for human involved collaboration. When multiple urgent events simultaneously happen at different places with unexpected requirements, administrators should schedule workforce under consideration of system and user context. In this paper, we investigate the problem of workforce management for urgent collaboration with context-awareness. We study the Feasible Workforce Assignment Problem (WAP) that determines whether a system state satisfies the requirements of a given multiple-event scenario. We present formal definitions of this problem and an efficient polynomial solution, as well as a case study.

As continuing work, we are investigating in the search for an optimal solution for this problem. We are also planning to consider more sophisticated context information, such constraints on specific users or tasks. It also would be interesting to take these considerations into account when solving WAP. In practice, there may not exist enough qualified users in an organization for the event requirements. We would like to investigate in solutions for such cases. Another future research direction is to investigate the secure delegation problem, which also handles exceptions and alternatives.

Acknowledgment

Part of this work is supported by the National High Technology Research and Development Program (863 Program) of China (2006AA01A113) and by the Science Foundation of Shandong Province Project (Y2008G28).

References

[1] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *IEEE Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[2] R. K. Thomas and R. S. Sandhu, "Task-based authorization controls: A family of models for active and enterprise-oriented autorization management," in *Proceedings of the IFIP TC11 WG11.3 Eleventh International Conference on Database Securty XI*, London, UK, 1998, pp. 166–181.

[3] W. Tolone, G.-J. Ahn, T. Pai, and S.-P. Hong, "Access control in collaborative systems," *ACM Computing Surveys*, vol. 37, no. 1, pp. 29–41, 2005.

[4] S. Oh and S. Park, "Task-role-based access control model," *Information Systems*, vol. 28, no. 6, pp. 533–562, 2003.

[5] Y. Sun, M. Farwick, and D. K. Chiu., "Constraint-based authorization management for mobile collaboration services," in *Proceedings of 2009 IEEE Congress on Services*, Los Angeles, USA, 2009, pp. 22–29.

[6] C. A. Ardagna, M. Cremonini, E. Damiani, S. D. C. di Vimercati, and P. Samarati, "Supporting location-based conditions in access control policies," in *Proceedings of ACM Symposium on Information, computer and communications security*, 2006, pp. 212–222.

[7] I. Ray and M. Kumar, "Towards a location-based mandatory access control model," *Computers and Security*, vol. 25, no. 1, pp. 343–358, 2006.

[8] I. Ray and M. Toahchoodee, "A spatio-temporal role-based access control model," in *Proceedings of the 21th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*. Springer, 2007, pp. 211–226.

[9] M. L. Damiani, E. Bertino, B. Catania, and P. Perlasca, "Geo-rbac: A spatially aware rbac," *ACM Transaction on Information System Security*, vol. 10, no. 1, pp. 1–42, 2007.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein., *Introduction to Algorithms*. The MIT Press, 2002.

[11] Y. Sun, B. Gong, X. Meng, Z. lin, and E. Bertino, "Specification and enforcement of flexible security policy for active cooperation," *Elsevier Information Sciences*, vol. 179, no. 15, pp. 2629–2642, 2009.