FORMING THE OPTIMAL TEAM OF EXPERTS FOR COLLABORATIVE WORK

Achim P. Karduck and Amadou Sienou
University of Applied Sciences, Furtwangen
Robert-Gerwig-Platz 1, 78120 Furtwangen, Germany

Abstract The "networked economy" challenges organizations to reconsider their business models. The aim of our work is to assist organizations dealing with challenges concerning team building. In order the deal with questions related to the formation of the best team, we have defined team formation as the process of allocating the set of experts that best fulfils team requirements.

Focusing on how to support the selection of experts for collaborative work, we have formulated team formation as a constraints satisfaction optimization problem solved by extending the iterated hill-climbing search strategy.

Keywords: Resource allocation, constraints satisfaction optimization, Decision support, Ad-hoc team formation, Collaborative work

1. Introduction

Suppose *SAM & associates systems, Inc.,* an organization of experts, is member of a worldwide heterogeneous network of financial consultants. Michael M., employee of this company, is expert on insurance questions. The consultant is now in conference with a customer in order to process questions concerning insurances. Because of the complexity of the problem, Michael M. has decided to connect to the headquarter in order to have support in some fields. In this case, the later have to form a team of experts that will assists him. For this purpose, an environment for Computer Supported Collaborative Work (CSCW) is required. It should allow the (1) creation of a shared workspace, provide (2) assistance for the team formation process, support (3) content management and enable (4) community support for synchronous and asynchronous collaboration processes. These requirements lead to scenarios for knowledge intense services that require seamless and spontaneous use of expertise in the back-office of an organization.

In this paper, we will explain how to form the best team in order to support this kind of scenarios. Our approach to team formation is based on Intellectual Asset Management (IAM). Intellectual Assets are the tacit assets of an orga-

nization, such as expert profiles or the results of an advice process. Derived from Modern Portfolio Theory (Fabozzi and Markowitz, 2002), the target is to optimize return versus risk of an asset allocation strategy.

2. Forming teams

2.1 What is team formation

As supported in to the group development theory (Tuckman and Jensen, 1977), a team goes through the phases forming, storming, norming, performing and adjourning to get performed while producing results. During the "forming" step, members are brought together for the first time for a particular task.

We call, team formation the process anterior to the forming stage of the group development theory. It consists in the identification and the selection of candidates. The decision to select a candidate is based on how well his/her profile fulfills the requirement specification. This supposes the existence of criteria to be considered while analysing the profiles. Since the focus is on the formation phase, factors essential to start the "forming" stage of team development are those that really matter. In the literature, some factors have been empirically evaluated or applied to other teams (Anderson, 1996; Deborah and Nancy, 1999; Lipnack and Stamps, 2000; Tuckman and Jensen, 1977; Schutz, 1955). In previous investigations we have considered the following factors (Karduck and Sienou, 2004):

Factors	Description
Competencies	Skills and experiences of experts
Interest	Skills that experts want to improve in teams
Risk	The financial risk of having an expert in a team
Availability	Time frame, when an expert is available for a team
Commitment	Financial attachment of an expert to a team
Budget	Amount of money available for a project
Constraints	Constraints related to skills, availability and budget

Table 1. **Factors affecting team formation.**

2.2 A model for team formation

In order to carry out a project, this one is subdivided into tasks able to be carried out by single experts. A task, in order to be performed, requires a set of competencies and interests. Experts are entities with interests and competencies who are looking for positions. Team brokerage consists of finding the right expert for a task. Assigning a set of tasks defined in the context of a

project to a set of experts is a Resource Allocation Problem (RAP). There are different approaches to solve RAP. Constraint programming is one of the successful approaches (Tsang, 1993). Key concepts of constraint programming are variables, their domains and constraints. Here, constraints are restrictions on the values to which a variable may be instantiated. The goal is to find an instantiation of all variables that satisfy all constraints.

Team formation is the Constraint Satisfaction Problem (CSP) (*Z, D, C*) defined as follows:

Variables Z: $\mathbf{Z} = \{z_1, .., z_n\}$ is the set of variables, each standing for a task which should be assigned to an expert.

Domains D: $\mathbf{D} = \{d_1, .., d_n\}$ is the set of domains of values to which variables z_i can be instantiated. Let us call elements of d_i instances denoted *I*. An Instance is the assignment of a task to an expert. Instances are conceptualized as follows:

- **Budget.** Amount of money planned for the task to be assigned; denoted *B*(*I*).
- **Cost.** A value representing the cost of assigning a task to an expert. Denoted *C*(*I*), this value is the product of the hourly wage required by the expert and the number of hours planned for the task.
- **Level of Commitment.** The *commitment breaking cost* is the amount that a partner will have to pay if he leaves the organization before the achievement of goals (Petersen and Divitini, 2002). The level of commitment is denoted $L(I) \in [0, 1]$.
- **Performance.** A value, denoted $P_{instance}$ that expresses the performance value of an instance.

Constraints C: **C** is a set of constraints in *Z* and *D*. Constraints have been classified into availability constraint, position constraint, instance constraints and team constraints.

- **Availability constraint.** An expert is eligible for a team if he/she is available during the executing tune of the task for which he/she is applying.
- **Position constraint.** An expert applying for a task will be considered only if the position posted in the context of this task is the one that he/she is looking for.
- **Instance constraints.** These constraints are restrictions to the level of interest, level of skill and experience. An expert is considered having an interest, skill or experience only if his/her levels of skills or experience are at least equal to the level defined in the constraints.

- **Team constraints.** In contrast to instance constraints, which refer only to single experts, team constraints affect the whole team.
  Let us introduce a binary variable $x_I \in \{0,1\}$ expressing whether an instance is activated; i.e. if the task z_i is assigned to a given expert, the value $x_I = 1$; otherwise $x_I = 0$. Based on the properties of single resource allocation we have defined the following constraints:

  $\mathbf{C_1}$. The total budget planned for a project should not be exceeded:

$$\sum_{i}^{n} \sum_{I \in z_i} C(I) \times x_I \leq \sum_{i}^{n} \sum_{I \in z_i} B(I) \times x_I \tag{1}$$

  For convenience, let us define the quotient

$$\Delta_{budget} = \frac{\sum_i^n \sum_{I \in z_i} C(I) \times x_I}{\sum_i^n \sum_{I \in z_i} B(I) \times x_I} \tag{2}$$

  and specify this constraint as follows:

$$\Delta_{budget} \leq 1 \tag{3}$$

  $\mathbf{C_2}$. All tasks must be assigned and each only once:

$$\forall z_i \in Z, \sum_{I \in z_i} x_I = 1 \tag{4}$$

Like any constraint satisfaction problems, this one will also have one, none or multiple solutions $X = \langle I_1, ..., I_n \rangle$. A solution is a n-dimensional vector representing a team. A solution is valid if both constraints C_1 and C_2 hold. The search space has a size of $\prod_{i=1}^{n} |d_i|$. Therefore, we need to support the selection of a team by introducing the concept of team performance value *P,* which maps each solution *X* to a value *P*(*X*). The value *P*(*X*) depends on single performance values of the experts involved in the team *X*.

3. Evaluating performance values

Selecting an expert for a given task is a decision problem depending on multiple criteria. The decision is based on the performance of an instance which is an aggregate value of the factors cost, synergy, skills, experiences, commitment and risk interpreted as the following non-dependant criteria:

- **Cost.** The cost of assigning a task to an expert denoted i is $C(i)$. In order to express this factor in term of positif values, we will consider the

function Δ_{cost} expressing how expensive is an expert in comparison with the worst case.

- **Level of commitment.** Value defined in the previous section as *L(I).*
- **Synergy.** The *value of synergy* is the similarity V_s between candidate interests and the interests required for the task in question. Here, interest is defined as list of attributes-values pairs with a level of interest.
- **Competency (skills and experience).** Let V_c be the aggregate value of the competency of an expert. This value is defined as the similarity between his/her competencies with the ones required for the task. The competency of an expert is the set $\{(A_i, \ell_i, expe_i), i = 1...n\}$ where A_i is an attribute-value pair representing the skill, ℓ_i the level of skill and $expe_i$ the experience expressed in number of months.

Let $\omega_i, i = 1...4$ be weighting factors representing the initiator's relative preferences for the criteria Δ_{cost}, V_s, V_c and $L(I)$ respectively. We define the performance $P_{instance}$ of an instance I as follows:

$$P_{instance}(I) = \sum_{i=1}^{4} \omega_i \times I_i^{value} \tag{5}$$

where

$$I_{i=1..4}^{value} = \langle \Delta_{cost}, V_s, V_c, L(I) \rangle$$

This value expresses "how well" the profile of an expert fulfills the requirement specification of a given task. Similarly, we define the performance value $P(X)$ of the whole team as follows:

$$P(X) = \sum_{i=1}^{n} \omega_i \times \sum_{I \in z_i} P_{instance}(I) \times x_I \tag{6}$$

Here $\omega_i \in [0, 1]$ is a weight representing the relative importance of the instance I and $\sum_{i=1}^{n} \omega_i = 1$. At this stage of the formation process, the main problem to deal with is to find the team with the highest performance value so that all tasks are assigned to exactly one expert and the total cost does not exceed the total budget.

4. Finding the optimal team

4.1 Problem statement

Finding the optimal team is the following single resource allocation and Constraint Satisfaction Optimization Problem (CSOP)(see eq. 6 for $P(X)$):

$$\begin{array}{ll} find & X = \langle I_1, ..., I_n \rangle \\ s.t. & maximize\ P(X) \\ subject\ to & \\ & \Delta_{budget}(X) \leq 1 \\ & \forall z_i \in Z,\ \sum_{I \in z_i} x_I = 1 \\ & \forall I \in z_i,\ x_I \in \{0, 1\} \end{array} \quad (7)$$

Notice that $\forall I \in z_i, i = 1...n, \omega_i \geq 0$ and ω_i is constant.
Since $\sum_{I \in z_i} P_{instance}(I) \times x_I \geq 0,\ \forall\ I \in z_i$, the value $P(X)$ is maximal if the values $P_{instance}(I)$ are maximal; i.e. the team performance is maximal if tasks are assigned always to experts having the highest performance value. We claim that it is possible to find the best team without executing an exhaustive search. For this purpose, consider the following search problem.

Objective. The objective consists in maximizing the team performance *P*, not exceeding the total budget, and assigning each task only once. Since it is algorithmically possible to satisfy the last constraint ($\forall z_i \in Z,\ \sum_{I \in z_i} x_I = 1$) by instantiating each variable exactly once, this one is no longer relevant to the solution if the control of the algorithm forces single allocations. The objective consists henceforth in maximizing the team utility without exceeding the total budget.

Formally, a state $X = \langle I_1, ..., I_n \rangle$ is a solution if

$$maximized_k(P(X)) \wedge C(X) = true$$

where the constraint *C* is $C(X) : \sum_i^n C(I_i) \leq \sum_i^n B(I_i)$.
Here $maximized_k(P(X)) = true$ if *X* is a solution and $P(X) \geq P(x_k)$ for the k first potential solutions.

Representation. We have defined a state as a vector of instances. A state $X = \langle I_1, ..., I_n \rangle$ represents a team consisting of the I_i^{th} candidate of the i^{th} task.

Evaluation function. A state X_1 is better than another one X_2 if $P(X_1) > P(X_2)$. A state X_1 is however a solution only if the *cost constraint* is satisfied; i.e. $\Delta_{budget}(X_1) \leq 1$. Notice that this evaluation metric does not provide any information about the satisfiability of the *cost constraint.*

Operators. Let us suppose that experts are ranked according to the decreasing order of the values of $P_{instance}$. In case that the best team is not a valid solution, one will examine the next best team. This one is formed by replacing exactly one candidate for a task by the next best candidate for the same task. This process is a 1-flip operation used to expand the search space. We say that a state X' is a neighbor of the state *X* if the former is obtained by flipping the value of only one variable of *X*.

4.2 Search strategy

Given that we intend to maximize the profit $P(X) = \sum_{i=1}^{n} \omega_i \sum_{I \in z_i} P_{instance}(I) \times x_I$ and given that the operands $\omega_i \geq 0$ and $\sum_{I \in z_i} P_{instance}(I) \times x_I \geq 0$ for all instances $I \in z_i$, the best state is without any doubt the set of the best instances; i.e. $max(P(X)) = \sum_i \omega_i \times max(\sum_{I \in z_i} P_{instance}(I) \times x_I)$. The best state is however not necessarily a valid solution since the *cost constraint* must be satisfied. In case that the best state is not a valid solution, one will examine the next best one, and so on, until a valid solution is found.

In order to find solutions without systematically searching the whole space, we have considered the iterated hill-climbing strategy. The Hill-climbing search is a local search strategy, which starts from an initial point of the search space. While iterating, the strategy selects a new point from the neighborhood of the current point, and replaces the current point by the new one if this one is better than the current. If a maximum number of iterations is reached or no more improvements are possible, the algorithm terminates. The iterated hill-climbing consists in searching the space until a given maximum number of iterations is reached (Michalewicz, 1999). The basic concepts of this algorithm are applicable to our search problem.

Let us consider the table 2. Let $\Omega = \langle 0.32, 0.26, 0.42 \rangle$ be weights expressing the relative preference for the variables z_1, z_2 and z_3 respectively. The state $\langle 0, 1, 1 \rangle$ is interpreted as the instantiation of the variable z_1, z_2, z_3 to the instances having the indexes 0, 1, 1 respectively. This state is actually the team formed by selecting the best candidate for $task_1$ and the second best for $task_2$ and $task_3$. The numbers in columns 2, 3 and 4 are performance values of the corresponding instances (candidates' scores regarding to tasks). The value at $(z_1, 0) = 0.528$ is for instance the score of allocating $task_1$ to the candidate at the index 0.

	Variables and domains		
	z_1	z_2	z_3
0	0.528	0.618	0.53
1	0.471	0.593	0.452
2	0.443	0.404	0.351
Ω	0.320	0.260	0.420

Table 2. **Scenario: variables and domains**

The best initial state is $P(000) = 0.552$. The iterated hill-climbing algorithm suggests the generation of neighbors and the selection of the best one. The current state becomes therefore better with each iteration. As outlined in table 3, in this case, neighbors of the initial state $P(000) = 0.552$ are (010), (100), (001). Let us suppose that the initial state is not a valid solution (i.e. it does not satisfy the *cost constraint*), the next best neighboring state to be examined is therefore $P(010) = 0.546$. If the state (010) in turn is not a solution, the algorithm has to select its next best neighbor which is (000) since the 1-flip operator expanding states always chooses the best values. Given that the state (000) has been already visited, the algorithm sticks around this one. In order to avoid this, we will introduce a *tabu list* which is a memory where already visited states are stored (algorithm 3). In the reminder of this section, the expressions "tabu state" or "is tabu" means that the state is in the *tabu list.*

Let us suppose once again that the current state $P(010) = 0.546$ does not satisfy the *cost constraint;* according to the iterative hill-climber, its best neighbor $P(110) = 0.528$ should become the current state. However, notice that there is one already computed state neighboring the predecessor of $P(010) = 0.546$ which is better than $P(110) = 0.528$ (the best neighbor of the current state). Actually, the neighbor $P(100) = 0.534$ of the initial state (000) is better than $P(110) = 0.528$. Given that we intend to visit best states first, the next state should be (100). Since the classic iterated hill-climbing algorithm does not support this case, we will introduce a concept similar to the best-first search which consists in defining a second memory called *open list* (algorithm 3) to store generated neighbors of the current state. The new algorithm should henceforth always select the best state of the *open list.* The current state becomes therefore $P(100) = 0.534$ while $P(110) = 0.528$ becomes the next in case that this former has to be invalid. The state next to $P(110) = 0.528$ is $P(200) = 0.525$ which has only two neighboring states since all states able to be its third neighbor are in the *tabu list.* The algorithm terminates if all states are in the *tabu list* or the objective of finding the best of the next ***k*-states** satisfying the *cost constraint* is reached. The whole algorithm called *team-former* is listed in algorithm 3. Here the best team is the state (000) that corresponds to the instances $\langle I_0, I_1, I_1 \rangle$; i.e. the best candidate for $task_1$, the second best for $task_2$ and $task_3$.

Path	State	Neighbors		
1	(000)	P(010)=0.546	P(100)=0.534	P(001)=0.519
2	(010)	P(110)=0.528	P(011)=0.513	P(020)=0.497
3	(100)	P(110)=0.528	P(200)=0.525	P(101)=0.501
4	(110)	P(210)=0.519	P(111)=0.495	P(120)=0.478
5	(200)	P(210)=0.519	P(201)=0.492	
6	(001)	P(011)=0.513	P(101)=0.501	P(002)=0.477
		...		
27	(222)			

Table 3. **Sample states and neighbors.**

Team-Former (Z, D, k):
  space: $Z = \{z_i, \quad i = 1...n\}, D = \{d_i, i = 1..n\}$
  tabu, *open*, *result*, *counter*
  $s_{best} \Leftarrow$ best state
  $s_c \Leftarrow s_{best}$
  while (*open* is not empty)
    add s_c to *tabu*
    if ($\Delta_{budget}(s_c) \leq 1$) /** s_c is a solution **/
      add s_c to *result*
      if (s_{best} not solution)
        $s_{best} \Leftarrow s_c$
      end if
      if (*counter* $\geq k$)
        return *result*
      end if
    end if
    if (s_{best} is a solution)
      counter $\Leftarrow$ *counter* $+ 1$
    end if
    open $\Leftarrow$ *open* $\cup$ *neighbors*(s_c) /** expands the space **/
    $s_c \Leftarrow$ best state from *open*
    remove s_c from *open*
  end while

To exemplify the *team-former* algorithm, consider Figure 1 illustrating the search path with the parameter $k = 10$. Points represent performance values of all states of the space whereby circles are performance values of the k-best solutions states. States which should be visited if no solutions were available are also represented by points. As outlined in figure 1, states with high performance values are visited first and the algorithm will stop at the 10^{th}. Given

that no state after the 10^{th} state is better than the best of the k-first, our outgoing assumption concerning the possibility to find the best solution without an exhaustive search of the state holds for this scenario. Notice that the strategy

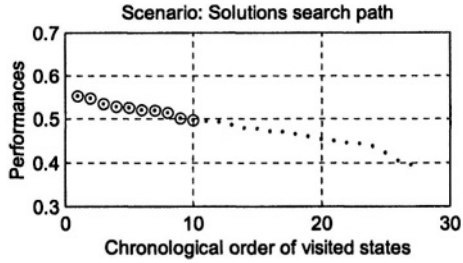

Figure 1. Scenario: team-former solutions and path

does an exhaustive search if no solution is available. In addition, if solutions have low performance values, they will be visited late. That is because the algorithm searches by decreasing the value of performance and believes that the great part of solutions is high performed. This is an optimistic approach to forming teams since we make the assumption that teams with hight performances may not exceed the total budget.

Comparing the *team-former* algorithm with the iterated hill-climber, it is necessary to notice some points.

The *team-former* algorithm does not repeat until a maximum depth of iteration is reached. In contrary, it runs until the k^{th} state after the first solution has been visited or the whole space has been expanded. When the domain is completely expanded, the algorithm becomes an exhaustive search. The concept of k-states solution is however similar to the iteration's depth of the iterated hill-climber. The factor k has not a particular role concerning the outcome of the search. It is rather a concept introduced in order to satisfy the statement (relative maximization) of the problem.

In contrast to the iterated hill-climber, which does not have any memory, the *team-former* is able to manage two lists in order to avoid local maxima, to prevent local maxima and to improve the search by generating and visiting only opened states. It encloses features from tabu-search and best-first search. Unlike *iterated hill-climbing,* the *team-former* strategy always select the best state by merging the neighborhood of the current state with all previously processed non-tabu states. In addition, each state is visited only once.

While having reasoned about performances of states, one should have noticed that the values *P* are decreasing from state to state instead of increasing as intended while formulating the problem as a maximization one. In fact, the ob-

jective is to find the solution(s) with the optimal performance in a given range of values (k) rather than absolutely maximizing the performance function.

5. Related Work

Concerning the area of the formation of teams, works related to our project are the ones from (Rub and Vierke, 1998; Petersen and Gruninger, 2000; Petersen and Divitini, 2002). In contrast to the first concept which supports the configuration of virtual enterprizes, we have emphasized the formation of virtual teams of humans. Both concepts share the aspects of optimal resource allocation. Unlike the second approach, where the virtual team initiator is the entity evaluating partners, we have adopted a brokerage approach. The approaches differs deeply in the metrics and the aggregation procedures used to process performances values.

6. Conclusion

The work on team formation support will be extended according to the *"3C Collaboration Model (Communication, Coordination, Cooperation)"* (Fuks and de Lucena; C. J., 2004). Especially coordination will be investigated in two aspects: *awareness* of the team members concerning the individual activities and *commitment* in terms of the delegation of individual responsibilities.

The model developed will be extended in the future concerning IAM with respect to risk/result-correlations, as applied in Modern Portfolio Theory in Finance (Fabozzi and Markowitz, 2002). We expect, that transferring these concepts will lead to new perspectives in requirement analysis and project management in general, in team formation in particular. One aspect we plan to investigate further is the correlation between the team size and the effectiveness of team work. Even we believe that this research question is difficult to quantify, we believe it to be an important measure due to the correlation of team size and coordination cost.

In the context of TeamBroker, we have (1) conceptualized factors and performance values affecting the formation of teams, (2) formalized constraints imposed by the project that a team has to carry out, (3) transformed the problem of forming teams into a resource allocation problem and (4) solved it by using methods of constraints programming. We have developed the Team-Former algorithm, which is an extension of the iterated-hill climbing search strategy. The algorithm guarantee to find the best solution first if any one exists.

A successful application of our concept to team formation requires the definition of performances metrics for competencies and interests by processing the similarity between vectors representing experts' competencies or interest and the one representing task requirements.

Because of the similarity between the team-former algorithm and the A* algorithm, in the future, we plan to discuss the team-former algorithm in relation to this one and to explain under which conditions the team-former becomes an A* algorithm.

References

Anderson, W. (1996). Human resource development challenges in a virtual organization. In *IEMC Proceedings: Managing the Virtural Enterprise,* Vancouver, B.C.: IEMC.

Deborah, L. D. and Nancy, T. S. (1999). *Mastering Virtual Teams: Strategies, Tools, and Techniques That Succeed.* 0787941832. Jossey-Bass Pub, San Francisco.

Fabozzi, F. J. and Markowitz, H. N. e. (2002). *The Theory and Practice of Investment Management.* Wiley, 1 edition.

Fuks, H.; Raposo, A. B. G. M. A. and de Lucena; C. J., P. (2004). Applying the 3c model to groupware engineering. *Monografias em Ciência da Computção,* (1).

Karduck, A. P. and Sienou, A. (2004). Teambroker: Constraint based brokerage of virtual teams. In *Proceedings of the sixth International Conference on Enterprise Information Systems (ICEIS 2004),* Porto, Portugal.

Lipnack, J. and Stamps, J. (2000). *Virtual Teams.* John Wiley & Sons, 2 edition.

Michalewicz, Z. / Fogel, D. B. (1999). *How to Solve It: Modern Heuristics.* 3540660615. Springer Verlag.

Petersen, S. A. and Divitini, M. (2002). Using agents to support the selection of virtual enterprise teams. In *Proceedings of Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002),* Bologne, Italy. AAMAS 2002.

Petersen, S. A. and Gruninger, M. (2000). An agent-based model to support the formation of virtual enterprises. In *International ICSC Symposium on Mobile Agents and Multi-agents in Virtual Organisations and E-Commerce (MAMA '2000),* Woolongong, Australia.

Rub, C. and Vierke, G. (1998). Agent-based configuration of virtual enterprises. In Holsten, A. e. a., editor, *Proc. of the Workshop on Intelligent Agents in Information and Process Management KI'98,* volume 9.

Schutz, W. (1955). What makes groups productive? *Human Relations,* 8:429–465.

Tsang, R. (1993). *Foundations of Constraint Satisfaction.* Academic Press.

Tuckman, B. and Jensen, N. (1977). Stages of small-group development revised. *Group and Organizational Studies,* 2(4):419–427.