Human resource load balancing based on Ant Theory for QoS management within an enterprise in a developing country

Roger Atsa Etoundi
Department of Computer Science
University of Yaoundé I
roger.atsa@uy1.uninet.cm

Marcel Fouda Ndjodo
Department of Computer Science
University of Yaoundé I
marcel.fouda@uy1.uninet.cm

Priso Essawe Ndedi
Department of Computer Science
University of Yaounde I

Ghislain Abessolo Alo'o
Department of Computer Science
University of Yaounde I
ghislainabess@yahoo.fr

RÉSUMÉ. Dans cet article, basé sur la théorie des fourmis, un protocole permettant l'équilibrage de charges entre les employés au sein d'une entreprise dans un pays en voie de développement est présenté, pour une amélioration de la qualité de service rendu aux bénéficiaires. Le papier termine avec la présentation des avancées obtenues après son application dans le processus de traitement dossiers de carrier des agents publics au Cameroun.

ABSTRACT. In this paper based on ant colony optimization, a protocol dealing with load balancing between employees of an enterprise in a developing country is presented for an efficient management of available resources for a better quality of service management. The paper ends by presenting the results obtained after its application in the processing of civil servant files in Cameroon.

MOTS-CLÉS : Equilibrage de charge entre personnels, gestion des workflows, Planification des ressources humaines, Pays en voie de développement, Qualité de service.

KEYWORDS: Human Load Balancing, Workflow Management, Workforce Planning, Developing countries, Quality of Service.

1. Introduction

Nowadays, a considerable number of organizations in developing countries are adopting workflow management systems to support their business processes. The current systems available manage the execution of workflow instances without any quality of service management on delivery deadlines [4]. This, in turn, fulfills customers' expectations and achieves customers' satisfaction. It is not sufficient just to describe the logical or operational functionality of activities and workflows. Rather, design of workflows, according to the context of developing countries must include the management of resources involved in the achievement of associated activities in order to guarantee the associated deadlines. The execution of a workflow instance is carried out by both machines and business process labors [1]. Each service whose plan has been proceeded is required to be achieved as soon as possible in order to deal with new arrival services or to predict the future [5]. Furthermore, an organization, to deal with the increasing number of services and the limited number of participants [3], has to assign associated activities to their labors without any control. As a result, some labors will be heavily loaded, lightly loaded or moderately loaded, while others will be completely idle. In order to deal with the human load balancing problem, the Ant Colony Optimization approach (ACO) [2] is used.

The rest of the paper is structured as follows: In Section 2, we present the Enterprise Modeling suitable for the management of the quality of service based on human actors. Section 3 deals with the Ant Theory Model. Section 4 presents the integration of the protocol in the human resource management process. Section 5 deals with the Correctness of the proposed protocol while Section 6 concludes the paper and highlights some future works.

2. Enterprise Modeling.

Based on the management of human actors for end users' satisfaction, the modeling takes into consideration tasks description, the enterprise workstations and the human actors.

2.1. Workstation

A workstation is defined as a set of tasks to be carried out by a human actor appointed at this position. According to the type of tasks to be performed in a given position, a certain number of skills are required. These skills should be enough such that all the associated tasks have the same chance to be executed. In our modeling, we do not accept workstation wk with an empty set of tasks i.e. $tasks(wk) \neq \varnothing$. Moreover, if wk denotes a workstation, *tasks(ws)* denotes its associated set of tasks.

2.2. Human Actor Model

The management of an employee within an enterprise is required to keep track of working time but also take into consideration the different schedules defined within them. When *Emp* represents an employee, *agenda(Emp)* denotes his diary, *maintime(agenda(Emp))* the list of time periods related to some dates where *Emp* can perform tasks, *schedule(agenda(Emp))* a list of jobs *Emp* is assigned. If *JB* is a job, *task(JB)* denotes the associated task, and *period(JB)* the related period. We require that all tasks contained in an agenda be scheduled according to the working time of the associated employee. This requirement is captured by the following constraints:

axiom

$(\forall$ **dr** : Diary, sh : Job • sh $\in$ **elems** schedule(dr) $\Rightarrow$

$(\exists$ dt : Date, p : TimeInterval • dt $\in$ **dom** maintime(dr) $\wedge$

p $\in$ **elems** maintime(dr)(dt) $\wedge$ isincluded(period(sh), p)))

Our modeling takes it into consideration by defining, for each labor, its associated skills. This is captured by the term *skills(Empl)*. The skill *Sk* itself is defined as a set of tasks, i.e. *tasks(Sk)*, that can be performed by a qualified labor.

2.3. Enterprise Model

Based on the human actor aspects and the load balancing issue, an enterprise org is viewed as:

- a set of labors labors(org) such that if lid denotes a labor identification then labors(org)(lid) denotes a specific employee,
- a set of workstations stations(org),
- the set of processing chains processingchain(org) where each processing chain defines a list of workstations that should be followed in order to meet a specific goal. We require that each workstation be in at least on the process chain of the enterprise. That is:
  axiom
  $(\forall$ wrk : WorkStation, org : Organization • wrk $\in$ stations(org) $\Rightarrow$
  $(\exists$ k:Nat, pch: Processchain• pch $\in$ processingchain(org) $\wedge$
  k $\in$ inds pth $\wedge$ wrk $\in$ elems pth))

Moreover, we also require that each work workstation included in a process chain be among workstations of the enterprise. This is captured by the following axiom

axiom

$(\forall$ org : Organization, i : Nat, pth : Processchain •

pth $\in$ processchain(org) $\wedge$ i $\in$ inds pth $\Rightarrow$ pth(i) $\in$ stations(org))

- the different positions of org is defined as a relationship between labors and workstations which is modeled as a map from labor identification to workstations such that, if lid is a labor identification then position(org)(lid) denotes the position of the employee labor(org)(lid). For efficient management or human resources, we do not allow an employee to have a position without having related competences, that is:
  axiom
  $(\forall$ org : Organization, wt : WorkStation, lid : LaborID•
  wt $\in$ dom position(org) $\wedge$ lid = position(org)(wt) $\Rightarrow$
  tasks(wt) $\subseteq$ dom competence(labor(lid)))

Moreover, we require that each workstation be assigned to an employee, and that each employee be appointed to only one position, this is captured by the following requirement:

axiom

$(\forall$ org:Organization• card stations(org) = card rng position(org)),

When MP denotes a map, rng MP denotes the range of MP

3. The Ant Theory Model

In the modeling, the skills required for the treatment of task are represented by the pheromone which is the substance secreted by ants to make the distinction among them. ***Type*** *Pheromone = Skill*. Three components are defined based on the ACO:

- *labor Ants Component* which defines the set of employees having the same skills and among which jobs can be exchanged. This type of ants is characterized by the given of a specific skill denotes by *colony(la)* and the associated list of employees *path(la)* having the skill *colony(la)*, where *la* denotes a given labor ant component;
- *Search Ants Component* which finds idle times and loads that can be delegated to another tally trades. This component is characterized by all the properties defined above, and the list *idletime(sa)* which determines the list of time interval *idletime(sa)(lid)* within which the employee, represented by his identifier lid, is able to perform jobs. The last characteristic is the list of jobs *load(sa)* that can be balanced to another employee. If *lbd* is an element of the list *load(sa)* then *snd(lbd)* denotes employee whose load is to be balanced, *rcv(lbd)* the eligible employee to receive the job *job(lbd)*.
- *Load Balance Ants Component* whose role is to update diaries of employees according to the information gathered by the Search Ants Component.

For the consistency of the model of our colony, we require that the following constraints be satisfied.

Requirement 3.1: All employees, whose information are keeping track within a workflow in the ministry must have the skill required in the workflow for efficient rooting of loads between employees. This requirement is captured by **colony(cd) ∈ dom skills(labor(rsid)))** where **cd** is a workflow and **rsid** an employee involved in the workflow **cd**.

Requirement 3.2: In our modeling, one should ensure that employees of the same workflow in the Ministry have the same skill. This requirement is captured by **colony(a) = colony(colony(cid)** where **colony(cid)** denotes the workflow defined by its identifier **cid** and **colony(colony(cid))** the skill of the colony **colony(cid).**

Definition (Nest): Lets *Lid* be a labor identification, a nest denoted by *Nt* is defined by the different loads *loads(Nt)(Lid)* of each employee , the list of idle time interval *idletime(Nt)(Lid)*, and the different colonies *colonies(Nt)* that constitute *Nt*.

We require that all employee involved in the treatment of tasks belongs to at least one workflow in order to ensure that his load will be checked during to load balancing process. This also guarantees the quality of service waited by stakeholders. That is $\forall$ *nt:Nest, cid: ColonyID* • ***dom*** *load(colonies(nt)(cid))* $\subseteq$ ***dom*** *loads(nt)*.

4. Integration of the Protocol in the Human Resource Management Process

4.1 Data Collection

To facilitate the job of the Manager, employees are regrouped according to their skills. Therefore, an employee assigned to a specific post should have the necessary skills to process all files that can be sent to that post. Moreover, for each post, a set of actions to be carried out denoted as a *profile*, by an employee assigned to that post.

4.2. Getting Loads Information

According to the type of task under consideration, the Head of Follow Up Unit after defining the set of employees having the necessary skill for their performance, the next action he undertakes consist of determining for each of them the associated load. This action will allow him to define when possible the employees who are overloaded and those who are underloaded. This information will be used in order to balance loads among these employees according to the time constraints associated to each file that is concerned.

Definition 4.1

Let *Nt* and *Org* denote respectively a nest and an organization, the second step φ_{12} of the protocol defined by the actions of labor ants consists in the definition of the nest Nt_{12} and the enterprise Org_{12} such that, If *A* denotes the set of load balancing data of a search ant *sa* just before reaching the labor lid and *B* the set of load balancing data after leaving lid, *C* the set of idle time interval of a labor *lid*□ just before *sa* reaches lid, *D* the set of time interval of *lid*□ just after *sa* leaves *lid*, the following properties are satisfied.

$$\begin{cases} lbd \in (A \cup B) - (A \cap B) \Rightarrow rcv(lbd) = lib \vee snd(lbd) = lib \\ tm \in (C \cup D) - (C \cap D) \Rightarrow isfree(lid', tm) \vee WF \end{cases}$$

where WF is defined as follows

$$\begin{cases} \exists l, l': LBData,\ l \in loads(sa) \wedge l' \in loads(sa') \wedge rcv(l) \neq lib' \wedge \\ task(job(l)) = task(job(l')) \wedge snd(l) = snd(l') \wedge rcv(l') = lib' \wedge tm = period(job(l')) \wedge \\ \neg isfree(lib', tm) \wedge started(tm) < started(period(job(l))) \wedge idealtime(task(job(l))) \leq started(tm) \end{cases}$$

In the nest, there is another type of ants whose role is to take food to different queens. We call these Ants *Guard*. Each guard may move data such as load balancing data, data related to time. The model of a nest previously defined is extended by the associated set of guards. Guards do not go out of the nest as their role is to be near the queens. If *nt* denotes a nest, *gaurds(nt)* denotes its set of guards. In our protocol, guards will take data collected in the colonies and labors to the queens. In our model we consider two queens, the time queen to which idle time is associated and load balanced queen to which load transfer data is associated.

4.2.1. Time Synchronization

Let *nt* be a nest, *cid* a colony, *gid* a guard and *lid* a labor, the action of the *guard(gid)* when the list of idle time *idletime(colonies(nt)(cid))(lid)* of the labor *labor(lid)* is not empty, a time interval is taken from this list in order to be inserted in the list of time intervals that are considered to be free for *labor(lid)*. The adding process can be seen as extending of an existing idle time interval when two time intervals overlap. In this level of abstraction, we will not define in detail how this is done but, we only require that there exists a time interval *tm* in the nest *nt*□ obtained after the action of *guard(gid)* such that the considered time interval *idletime(guards(nt)(gid))* of the guard *guard(gid)* and *tm* overlap i.e. *overlap(tm, idletime(guards(nt)(gid)))*. This action of the guard *gid* within the colony *cid*, the nest *nt* and the labor lid is call time synchronization and is denoted by the term *synchtime(nt, gid, cid, lid)* which is equivalent to a nest *nt*◘ obtained after the action of the guard to *nt*, *cid* and *lid*.

Definition4.2 *(Completion of time synchronization)*

We say that the synchronization time is complete if there does not exist a labor *labor(lid)* in a given colony *colony(cid)* of the nest *nt* such that the list *idletime(colony(cid))(lib)* of idle time of *labor(lid)* is not empty.

4.2.2. Load Synchronization

Let *p* be a time interval defined in the job *jb* with *jb = job(load(guards(nt)(gid)))* and *p = period(jb)*, *rcv = rcv(load(guards(nt)(gid)))* defines the associated receiver, *snd = snd(load(guards(nt)(gid)))* the associated sender, if there exists a time interval *tm* such that $tm \in$ *elems idletime(nt)(lid)* and the associated time interval *p* is included in *tm* then the load *LD*, defined by *Ld = load(guards(nt)(gid))*, is added to the list of data to be used by load balancing ants in the updating of labor diary process described above. After adding this *LD*, its associated time interval is removed from the list of idle time intervals of the receiver while it is added to the list of idle time interval of the sender as previously described.

Definition 4.3

Let Nt_{12} and Org_{12} respectively denote a nest and an organization obtained after the completion of time synchronization, the third step φ_{13} of the protocol defined by the actions of search, consists in the definition of the nest Nt_{13} and the enterprise Org_{13} such that :

$$\begin{cases} (N\,t_{13}, Org_{13}) = \varphi_{13}(N\,t_{12}, Org_{12}) \\ Org_{12} = Org_{13} \\ elems\ loads(N\,t_{12})(cid)) \subseteq elems\ loads(N\,t_{13})(cid) \\ cid \in dom\ colonies(nt_{13}) \Rightarrow loads(colony(cid))(lid) = empty \end{cases}$$

When a load has been positioned for updating, a Load Balancing Ant takes it in order to perform the updating of the agenda. This makes it possible at each time to have load balanced among labors within an enterprise.

4.3. Load Transfer

Load Balancing Ants use information defined by the guards in order to update the agenda of the receiver and the sender of the load.

Axiom 4.1

Let *lba* a load balancing information, aid a given labor such that if *rcv(lb(lba)) = aid* then the job *job(lb(lba))* is to be inserted in the agenda of *labor(aid)* that is $job(lb(lba)) \in$ *elems schedule(agenda(labor(aid)))* and the associated period is marked as occupied in the agenda of *labor(aid)* that is *not f ree(labor(lid), period(job(lb(a))))*. If the employee is the sender, that is *snd(lb(lba)) = aid* then the job is removed from his agenda, that is $job(lb(lba)) \notin$ *elems schedule(agenda(labor(aid)))* and the associated period is let free in the agenda, that is *f ree(labor(lid), period(job(lb(a))))*.

4.4. Complexity

Let n denote the number of entries within an employee's diary, m the number of employees within an enterprise, p the number of skills required in order to meet customer's needs. The process applied by labor ants in building the path followed by the members of their colony is carried out in $n \times m$ steps. The search ants in order to get schedule information require $2 \times n \times m$, as they go over the list of idle ti me and the schedules of each employee. The action of load balancing ants is carried out in one step. The action of guards is carried out in $p \times n$ steps. The resulting complexity is defined by $n \times m + 2 \times n \times m + p \times n$ that is $n(3 \times m + p)$. In general the number of skills is defined by the following: $p = k \times m$, which gives $4 \times k \times m \times n$, thus the complexity is given by $O(n^2)$ in the worse case. However, if the number of entries within a diary is larger than the number of employees, the complexity is then defined by $O(n)$. Moreover, the actions defined in the protocol are based on search process, the protocol's complexity can be defined by $O(log_2(n))$.

Definition 4.4

Let Org and *Nt* respectively denote an enterprise and a nest, φ_k, with $1 \leq k$ and $k \leq 4$, the intermediate steps of the protocol □, □ is defined by the following : $\varphi = \varphi_4 \circ \varphi_3 \circ \varphi_2 \circ \varphi_1$ with $\varphi(N t, Org) = \varphi_4 \circ \varphi_3 \circ \varphi_2 \circ \varphi_1(N t, Org)$.

---

5. Correctness of the Protocol

In order to check the correctness of the proposed protocol, we first need to fix our understanding about some concepts that are commonly used in the load balancing field. In the enterprise modeling and the human actors management, these concepts will not have exactly the same meaning. Among them are the underload and overload concepts.

Definition 5.1 *(Overloaded labor)*

Given an enterprise org and an employee lid, the labor *labor(lid)* is said to be overloaded if one can find another employee *lid*□ such that according to the time constraints among tasks of a given activity defined in the enterprise to meet a defined customer goal, the employee *labor(lid*□*)* is available to perform the task *tg* much early without violating the associated constraints mentioned above.

Definition 5.2 *(underloaded labor)*

Given an enterprise *org* and an employee *lid*, the labor *labor(lid)* is said to be underloaded if one can find another employee *lid*□ in org such that the performance of a given task of *labor(lid*□*)* can be shifted to *labor(lid)* within an earlier time interval without violating the associated time constraints.

value

underloaded : Nest × LaborID × Shift □ **Bool**

underloaded(nt, lid, jb) □ (∃ lid□: LaborID • {lid, lid□} ⊆ **dom** idletime(nt) ∧

jb ∈ **elems** schedule(agenda(labor(lid□))) ∧

overloaded(nt, lid□, jb) ∧ available(lid, jb, nt))

Where the term *available(lid, jb, nt)* means that there exists two time intervals *tm* and *tn* such that *tn* contains *tm* i.e. *isincluded(tm, tn)*, *tn* ∈ *idletime(nt)(lid)*, the labor *abor(lid)* has the necessary competence to execute the task *task(jb)* within the time interval *tm*, without violating the time constraints of *task(jb)*, which is earlier than *period(jb)* first defined as performance period.

6. Conclusion and Future Works

In this paper we have defined a generic protocol which brings out the balancing of loads among employees in an enterprise in developing countries. This protocol has been modeled based on the Ant Colony Approach. We have considered various types of ants according to the role each has to play in the load balancing process: Labor Ants have been used in order to determine the set of employees having the same skill and who can exchange loads, search ants are used in defining the senders and receivers for loads that are transferred; load balancing ants are used to update the agenda of employees involved if the process, and the guard ants which validate data defined by the labor and search ants based on the queen concept. The protocol is completed by defining its correctness property which is defined after fixing the meaning of well known concepts such as overload and underload agent.

The proposed protocol has been integrated in the computerized system of integrated management of State personnel and the payroll to increase the quality of service delivered to civil servants. The system is actually used in 30 ministries in Cameroon and has significantly increased the quality of the service as files are now processed in a reasonable time since 2004. As such, in the Ministry of Public Service and Administrative Reform, the number of files processed is increasing over the years regarding the file received as mentioned in this statistics diagram.

We have noticed that tasks performance does not follow any order. Therefore, the next step in this work, which is in progress, is the integration of tasks priority such that only the most prioritized task is performed before tasks of less priority.

7. References

[1] Van Oyen, M. P. (2004), '*Agile workforce evaluation: a framework for crosstraining and coordination*', IIE Transactions.

[2] Dorigo, M. & Socha, K. (2007), *An introduction to Ant Colony Optimization, Technical Report TR/IRIDIA/2006-010*, Universite libre de bruxelles.

[3] Etoundi, R. A., Ndjodo, M. F., Monessa, M. A. & Zobo, E. P. (2006), '*Feature-oriented workflow modelling based on enterprise human resource planning*', Business Process Management Journal 12, 608 – 621.

[4] Jorge Cardoso, Amit Sheth, J. M. J. A. & Kochut, K. (2004), '*Quality of service for workflows and web service processes*', Web Semantics: Science, Services and Agents on the World Wide Web 1, 281–308.

[5] Joseph, M. (1996), *Real-time systems Specification, Verification and Analysis*, Prentice Hall. This book presents in the chapter 4, dynamic priority scheduling.