Anul XLIX 2004

S T U D I A
UNIVERSITATIS BABEŞ-BOLYAI

INFORMATICA

1

Redacţia: 3400 Cluj-Napoca, str. M. Kogălniceanu nr. 1 Telefon 405300

SUMAR – CONTENTS – SOMMAIRE

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

STOCHASTIC OPTIMIZATION OF QUERYING DISTRIBUTED DATABASES III. EVOLUTIONARY METHOD VERSUS CONSTRUCTIVE METHOD

D. DUMITRESCU, C. GROŞAN, V. VARGA

Abstract. The stochastic query optimization problem for multiple join - join of p relations, stored at p different sites - leads to a special nonlinear programming problem. General optimization problem can be solved by using evolutionary methods. The problem of four joins is a particular case of the general query optimization problem. This particular problem is solved applying the Constructive Algorithm from Part II and the Adaptive Representation Evolutionary Algorithm. The results obtained by applying these two methods are compared. Two sets of constant values are used in this comparison.

Keywords Distributed Databases, Query Optimization Problem, Genetic Algorithms, Evolutionary Optimization, Adaptive Representation.

1. Introduction

The general stochastic optimization problem for the join of p relations, stored at p different sites of a distributed database was presented in Part I of the paper. This optimization problem in distributed databases leads to a constrained nonlinear programming problem. In Part I a theorem, which proves, that nonlinear optimization problem has at least one solution is stated. In Part II a constructive method for solving the nonlinear programming problem is proposed.

In this Part of the paper the optimization problem is solved using an evolutionary method presented in Dumitrescu, Grosan and Oltean (2001), Grosan and Dumitrescu (2002). Section 3 describes the evolutionary technique called *Adaptive Representation Evolutionary Algorithm* (AREA). In section 4 the results obtained

Received by the editors: June 6, 2004.

2000 *Mathematics Subject Classification.* 68P15, 68T99.

1998 *CR Categories and Descriptors.* C.2.4 [**Computer Systems Organization**]: Computer-Communication Networks – *Distributed Systems*; H.2.4 [**Information Systems**]: Database Management – *Systems*.

by applying these different approaches are presented. Two sets of constant values are used in these experiments. The solutions obtained by the two approaches are very close. The CPU time required for solving the optimization problem by using evolutionary algorithm is less than the CPU time required by the constructive method. Considered stochastic model is compared with a popular heuristic method (Section 5).

2. Solving problem (P_p) using an evolutionary algorithm

In this section an evolutionary technique used for solving problem (P_p) is proposed.

Let us denote $g_i = x_i + x_{i+1}$.

General problem (P) can be reformulated as the following constrained optimization problem:

$$(P') \begin{cases} \text{minimize } y \\ \quad \text{subject to:} \\ y > 0, \\ f_i(x) \leq y, \ i = 1, \ldots, p, \\ g_i(x) - 1 = 0, \ i = 1, \ldots n, \\ x = (x_1, \ldots, x_n). \end{cases}$$

The evolutionary method for solving problem (P') implies the next steps:

Step 1. Determine maximum from the p functions f_1,..., f_p.

Let us denote by

$$f^*(x) = \max(f_i(x)), x \in X.$$

Step 2. Minimize the function f^* by using an evolutionary algorithm.

In this case, the fitness of the solution x will be:

$$\begin{aligned} eval(x) &= f^*(x) \\ &= \max_{i=1,\ldots,n} f_i(x), \ x \in X. \end{aligned}$$

Let us denote by x^* the obtained minimum.

Step 3. The solution y of the problem can be obtained by setting

$$y = x^* + \varepsilon,$$

where $\varepsilon > 0$, is a small number.

Remark. The constraints g_i were treated by considering each x_i, i not odd, as being $1 - x_i$.

The advantage of applying an evolutionary technique for solving problem (P') is that the involved function f^* is not necessary effectively computed. Only the values of function f^* for the candidate solution are needed.

3. Evolutionary techniques for solving problem (P)

An evolutionary algorithm called *Adaptive Representation Evolutionary Algorithm* (AREA) is proposed for solving problem (P'). AREA technique will be described in what follows.

3.1. **AREA technique.** The main idea of AREA technique is use a dynamical encoding allowing each solution be encoded over a different alphabet. This approach is similar to that proposed in Kingdon and Dekker (1995). Solution representation is adaptive and may be changed during the search process as the effect of mutation operator.

3.2. **AREA representation.** Each AREA individual consists of a pair $(x,\ B)$ where x is a string encoding object variables and B specifies the alphabet used for encoding x. B is an integer number such that $B \geq 2$ and x is a string of symbols over the alphabet $\{0,\ 1,\ \ldots,\ B-1\}$. If $B = 2$, the standard binary encoding is obtained.

Each solution has its own encoding alphabet. The alphabet over which x is encoded may be changed during the search process.

An example of AREA chromosome is the following:

$C = (301453,\ 6)$.

Remark. The genes of x may be separated by comma if required (for instance when $B \geq 10$).

3.3. **Search operator.** Within AREA mutation is the unique search operator. Mutation can modify object variables as well as the last chromosome position (fixing the representation alphabet).

When the changing gene belongs to the object variable sub-string (x – part of the chromosome), the mutated gene is a symbol randomly chosen from the same alphabet.

Example

Let us consider the chromosomes can be represented over the alphabets $B_2, \ldots, B_{30}$, where $B_i = \{0, 1, \ldots, i-1\}$.

Consider the chromosome C represented over the alphabet B_8:

$C = (631751,\ 8)$.

Consider a mutation occurs on the position 3 in the x part of the chromosome and the mutated value of the gene is 4. Then the mutated chromosome is:

$C_1 = (634751, 8)$.

If the position specifying the alphabet is changed, then the object variables will be represented using symbols over the new alphabet, corresponding to the mutated value of B.

Consider again the chromosome C represented over the alphabet B_8:

$C = (631751, 8)$.

Consider a mutation occurs on the last position and the mutated value is 5. Then the mutated chromosome is:

$C_2 = (23204032, 5)$.

C and C_2 encode the same value over two different alphabets (B_8 and B_{10}).

Remark. A mutation generating an offspring worst than its parent is called a *harmful mutation*. A constant called MAX_HARMFUL_MUTATIONS is used to determinate when the chromosome part represented the alphabet will be changed (mutated).

3.4. **AREA procedure.** During the initialization stage each AREA individual (chromosome, solution) is encoded over a randomly chosen alphabet. Each solution is then selected for mutation. If the offspring obtained by mutation is better than its parent than the parent is removed from the population and the offspring enters the new population. Otherwise, a new mutation of the parent is considered. If the number of successive harmful mutations exceeds a prescribed threshold (denoted by MAX_HARMFUL_MUTATIONS) then the individual representation (alphabet part) is changed and with this new representation it enters the new population.

The reason behind this mechanism is to dynamically change the individual representation whenever it is needed. If a particular representation has no potential for further exploring the search space then the representation is changed. In this way we hope that the search space will be explored more efficiently.

The AREA technique may be depicted as follows.

AREA technique

begin
Set $t = 0$;
Random initializes chromosome population P (t);
Set to zero the number of harmful mutations for each individual in $P(t)$;
while ($t <$ number of generations) **do**
$P(t+1) = \emptyset$;

```
for k = 1 to PopSize do
  Mutate the k^th chromosome from P (t). An offspring is obtained.
  Set to zero the number of harmful mutations for offspring;
  if the offspring is better than the parent then
      the offspring is added to P(t + 1);
    else
      Increase the number of harmful mutations for current individual;
      if the number of harmful mutations for the current individual =
            MAX_HARMFUL_MUTATIONS then
        Change the individual representation;
        Set to zero the number of harmful mutations for the current
          individual;
        Add individual to P(t+1);
       else
        Add current individual (the parent) to P(t+1);
      end if
  end if
end for;
Set t = t + 1;
end while;
end
```

4. Experimental results. Constructive algorithm versus AREA technique

In this section we consider two numerical experiments for solving problem (P_p) using Constructive Algorithm and evolutionary technique above described. Results obtained by applying AREA technique are compared with the results obtained by applying Constructive Algorithm (and after that, Refinement Algorithm).

The obtained results in the case of a communication network with a speed of $6 \cdot 10^4$ bps are presented. The model allows different transfer speed for the distinct connections. But in our experiment the transfer speed is assumed to be constant for each connection. In the Table 1 and Table 4 appear two cases and in every case the number of bits for every relation and the necessary time to transfer the relations through the network.

We have to approximate the size of B', where

$B' = A \bowtie B$

and the size of C', where

$C' = B' \bowtie C$.

	Number of bits	Transfer time
Relation A	8,000,000	133.33s
Relation B	4,000,000	66.66s
Relation C	10,000,000	166.66s
Relation D	5,000,000	83.33s
Relation B'	10,400,000	173.33s
Relation C'	25,600,000	426.66s

TABLE 1. Relation sizes and transfer times for Experiment 1.

The database management system can take these sizes from the database statistics. In our computation we ignore the local processing time, because it is unessential compared to the transmission time.

4.1. **Experiment 1.** Replacing the transfer times in formulas (3.1) of the Part I and ignoring the local processing time we obtain:

$T_{11}(B') = 66.66$,
$T_{12}(C') = 166.66$,
$T_{32}(C') = 166.66$,
$T_{13}(D') = 83.33$,
$T_{33}(D') = 83.33$,
$T_{53}(D') = 83.33$,
$T_{73}(D') = 83.33$,
$T_{21}(B') = 133.33$,
$T_{22}(C') = 173.33$,
$T_{42}(C') = 173.33$,
$T_{23}(D') = 426.66$,
$T_{43}(D') = 426.66$,
$T_{63}(D') = 426.66$,
$T_{83}(D') = 426.66$.

Let us consider the mean processing times at the four sites, given by the formulas 3.2 of the Part I.

AREA Parameters	
Population size	100
Number of generations	100
Mutation probability	0,01
Number of variable	14
Number of alphabets	30
MAX_HARMFUL_MUTATIONS	5

TABLE 2. Parameters used by AREA technique.

$$
\begin{aligned}
\tau_1 &= 66.66p_{0,11} + 166.66p_{0,11}p_{11,12} + 83.33p_{0,11}p_{11,12}p_{12,13},\\
\tau_2 &= 133.33p_{0,21} + 166.66p_{0,21}p_{21,32} + 83.33p_{0,21}p_{21,32}p_{32,53},\\
\tau_3 &= 173.33p_{0,11}p_{11,22} + 173.33p_{0,21}p_{21,42} + 83.33p_{0,11}p_{11,22}p_{22,33}\\
&\quad + 83.33p_{0,21}p_{21,42}p_{42,73},\\
\tau_4 &= 426.66p_{0,11}p_{11,12}p_{12,23} + 426.66p_{0,11}p_{11,22}p_{22,43} +\\
&\quad + 426.66p_{0,21}p_{21,32}p_{32,63} + 426.66p_{0,21}p_{21,42}p_{42,83}.
\end{aligned}
\tag{4.1}
$$

From (4.1) we obtain the next values for constants $c_1, \ldots, c_{14}$ of Equations (4.1) of the Part I:

$c_1 = 66.66,$
$c_2 = 166.66,$
$c_3 = 83.33,$
$c_4 = 133.33,$
$c_5 = 166.66,$
$c_6 = 83.33,$
$c_7 = 173.33,$
$c_8 = 173.33,$
$c_9 = 83.33,$
$c_{10} = 83.33,$
$c_{11} = 426.66,$
$c_{12} = 426.66,$
$c_{13} = 426.66,$
$c_{14} = 426.66.$

Parameters used within AREA technique are given in Table 2:

The results obtained by applying AREA technique and Constructive Algorithm (and Refinement Algorithm after that) are outlined in Table 3.

Transfer probabilities	**Solutions obtained by AREA**	**Solutions obtained by the CA + RA**
$p_{0,11}$	0,701	0,7
$p_{0,21}$	0,298	0,3
$p_{11,12}$	0,404	0,4
$p_{11,22}$	0,595	0,6
$p_{21,32}$	0,901	0,9
$p_{21,42}$	0,098	0,1
$p_{12,13}$	0,513	0,55
$p_{12,23}$	0,486	0,45
$p_{22,33}$	0,755	0,75
$p_{22,43}$	0,244	0,25
$p_{32,53}$	0,970	0,95
$p_{32,63}$	0,029	0,05
$p_{42,73}$	0,978	0,85
$p_{42,83}$	0,0213	0,15
Δ_1	106,251	106,372

TABLE 3. Solutions obtained by AREA technique and CA for the first set of constants considered.

Remark. Final result obtained by AREA technique and CA is very similar. Only CPU time is different: CPU time obtained by AREA technique is 0.05s, and the CPU time obtained by CA is 11 minutes.

4.2. **Experiment 2.** Consider the experimental conditions given in Table 4. The values of constants $c_1,\ldots,c_{14}$ obtained from this conditions are the followings:

$c_1 =$ 16,66,
$c_2 =$ 33,33,
$c_3 =$ 16,66,
$c_4 =$ 133,33,
$c_5 =$ 33,33,
$c_6 =$ 16,66,
$c_7 =$ 173,33,
$c_8 =$ 173,33,
$c_9 =$ 16,66,
$c_{10} =$ 16,66,
$c_{11} =$ 213,33,

$c_{12} = 213{,}33$,
$c_{13} = 213{,}33$,
$c_{14} = 213{,}33$.

The parameters used by AREA in this case are presented in Table 5:

The results obtained by applying AREA algorithm and CA + RA are presented in Table 6.

Remark. According to Table 6 the final solutions obtained by these two algorithms are very close. CPU time obtained by AREA technique is 0.05 s, less than the CPU time obtained by CA, which is 15 minutes. Evolutionary algorithms seem to be useful technique for practical optimization proposes.

5. Stochastic model versus heuristic strategy

In this section we compare our stochastic optimization model and a very popular transfer heuristic [see Özsu, P. Valduriez, 1999]. According to the transfer heuristic the smaller relation from the operands of a join is transfered to the other operand relation. A query against a database is executed several times (not only once). The proposed stochastic model takes it into consideration and tries to share the execution of the same query between the sites of the network.

We say a strategy is "*pure*" if the execution path of the query in the state-transition graph is the same in every case the query is executed. If the query is executed several times, one of the joins of the query is executed in every case by the same site, and this is valid for every join of the query.

In the following we compare the results of the stochastic model with the results given by a "*pure*" strategy.

In step 1 of Experiment 1 the transmission of relation B is chosen several times, because it's size is smaller than the size of relation A. Therefore the system undergoes transition from state s_0 in state s_{11} in 7 cases from 10.

	Number of bits	Time to transfer
Relation A	8,000,000	133.33s
Relation B	1,000,000	16.66s
Relation C	2,000,000	33.33s
Relation D	1,000,000	16.66s
Relation B'	10,400,000	173.33s
Relation C'	12,800,000	213.33s

Table 4. Relation sizes and transfer times for Experiment 2.

AREA Parameters	
Population size	100
Number of iterations	10000
Mutation probability	0,01
Number of variables	14
Number of alphabets	30
MAX_HARMFUL_MUTATIONS	5

TABLE 5. Parameters used by AREA algorithm.

Transfer probabilities	**Solutions obtained by AREA**	**Solutions obtained by the CA + RA**
$p_{0,11}$	0,778	0,75
$p_{0,21}$	0,221	0,25
$p_{11,12}$	0,75	0,75
$p_{11,22}$	0,249	0,25
$p_{21,32}$	0,962	0,875
$p_{21,42}$	0,037	0,125
$p_{12,13}$	0,75	1
$p_{12,23}$	0,25	0
$p_{22,33}$	0,996	0,875
$p_{22,43}$	0,003	0,125
$p_{32,53}$	0,891	0,25
$p_{32,63}$	0,108	0,75
$p_{42,73}$	0,771	0,875
$p_{42,83}$	0,228	0,125
Δ_1	39,7492	40,3232

TABLE 6. Solutions obtained by AREA technique and CA for the second set of constants considered.

>From state s_{11} states s_{12} and s_{22} are chosen in a balanced mode. This because the size of relation B' is approximately equal to the size of relation C.

This balance is not so evident from state s_{21}. This can be explained by the approximative character of our methods.

In the third step of query strategy, which is the join of relation D with the result relation C', nearly in every case relation D is chosen for transfer. This because the size of D is much smaller than the size of relation C'.

Consider the "*pure*" strategy (deduced from transfer heuristic): s_0, s_{11}, s_{12}, s_{13}. Every join is executed in site 1, and the necessary time is: 66,66s + 166,66s + 83,33s = 316,65s, which is much greater than the mean processing time given by the stochastic query optimization model (This time is 106,251s).

In Experiment 2 the situation is similar. As the size of relation B is smaller than in case of Experiment 1 the transfer of it is chosen more often than in case of Experiment 1.

In step 3 in most cases the transfer of relation D is chosen. The size of D is much smaller than the size of the result relation C'. A "*pure*" strategy in this case with the same heuristic may be s_0, s_{11}, s_{12}, s_{13}. The necessary time for this "pure" strategy in site 1 is: 16,66s +33,33s + +16,66s = 66,65s, which is greater than the mean processing time given by the stochastic optimization model. (This time is 39,7492s)

6. Conclusions

In this paper the stochastic model is extended to the join of four relations. These four relations are stored in four different sites. The stochastic query optimization problem in case of four relations leads to a constrained nonlinear programming problem. For solving this problem two different approaches are considered: a constructive (exhaustive) one and an evolutionary one. The results obtained by applying these two methods are very similar. The difference consist in CPU time: by considering evolutionary method for solving the problem the execution time is less than the running time obtained by applying the constructive method.

References

[1] C. J. Date (2000): *An Introduction to Database Systems*, Addison-Wesley Publishing Company, Reading, Massachusetts.

[2] R. F. Drenick (1986): *A Mathematical Organization Theory*, Elsevier, New York.

[3] P. E. Drenick, R. F. Drenick (1987): *A design theory for multi-processing computing systems*, Large Scale Syst. Vol. 12, pp. 155–172.

[4] P. E. Drenick, E. J. Smith (1993): *Stochastic query optimization in distributed databases*, ACM Transactions on Database Systems, Vol. 18, No. 2, pp. 262–288.

[5] D. Dumitrescu, C. Grosan, M. Oltean (2001): *A new evolutionary adaptive representation paradigm*, Studia Universitas "Babes-Bolyai", Seria Informatica, Volume XLVI, No. 1, pp. 15–30.

[6] C. Grosan, D. Dumitrescu (2002): *A new evolutionary paradigm for single and multiobjective optimization*, Seminar on Computer Science, "Babes-Bolyai" University of Cluj-Napoca.

[7] J. Kingdon,L. Dekker (1995): *The shape of space*, Technical Report, RN-23–95, Intelligent System Laboratories, Department of Computer Science, University College, London.

[8] S. LaFortune, E. Wong (1986): *A state transition model for distributed query processing*, ACM Transactions on Database Systems, Vol. 11, No. 3, pp. 294–322.

[9] T. Markus, C. Morosanu, V. Varga (2001): *Stochastic query optimization in distributed databases using semijoins*, Annales Universitatis Scientiarum Budapestinensis de Rolando Eötvös Nominatae Sectio Computatorica 20, pp. 107–131.

[10] M. T.Özsu, P. Valduriez (1999) : *Principles of Distributed Database Systems*, Prentice-Hall.

[11] R. Ramakrishnan (1998): *Database Management Systems* WCB McGraw-Hill.

[12] W. Rudin(1976): *Principles of Mathematical Analysis*, McGraw-Hill, New York.

[13] J. D. Ullman (1988): *Principles of Database and Knowledge-Base Systems,* Vol. I-II, Computer Science Press.

[14] V. Varga (1998): *Stochastic optimization for the join of three relations in distributed databases I. The theory and one application*, Studia Universitas "Babes-Bolyai", Seria Informatica, Volume XLIII, No. 2, pp. 37–46.

[15] V. Varga (1999): *Stochastic optimization for the join of three relations in distributed databases II. Generalization and more applications*, Studia Universitas "Babes-Bolyai", Seria Informatica, Volume XLIV, No. 1, pp. 55–62.

Department Of Computer Science, Faculty of Mathematics and Computer Science, Babes-Bolyai University, Kogălniceanu 1, 400084 Cluj-Napoca, Romania

E-mail address: {ddumitr, cgrosan,ivarga}@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

EVOLUTIONARY CLUSTERING USING ADAPTIVE PROTOTYPES

D. DUMITRESCU AND R. GORUNESCU

Abstract. Genetic chromodynamics has proven to be a very efficient metaheuristics for detecting multiple optima points.

It is our goal to show that it is extremely suitable in the field of clustering as well.

Keywords: clustering, evolutionary computation, genetic algorithms, genetic chromodynamics, multiple optima points, stepping stone, local interaction, weighted similarity measures.

1. Introduction

A genetic chromodynamics-based clustering method is proposed.

Genetic chromodynamics (GC) is not a particular evolutionary technique but merely a metaheuristics for maintaining population diversity and for detecting multiple optima. Its strategy is to form and maintain stable subpopulations that co-evolve and lead, at convergence, each to an optimum. It uses a variable sized solution population, a stepping stone search mechanism in connection with a local interaction principle, and a special operator for merging very similar individuals. Therefore the clustering mechanism we get by using these principles is very simple: each convergence point represents a cluster, no number of clusters are given in advance and we will obtain in the end both the optimal number of groups and the optimal classification.

2. Genetic Chromodynamics. Basic Ideas

Here we summarize briefly the main underlying principles of genetic chromodynamics metaheuristics[2]:

- the population size is variable;
- the sub-population structure is not predefined;
- there is a stepping stone mechanism;

Received by the editors: September 12, 2003.

2000 *Mathematics Subject Classification.* 62H30, 68T20.

1998 *CR Categories and Descriptors.* I.5.3 [Pattern Recognition]: Clustering – *Clustering algorithms*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *Heuristic methods*.

- the local interaction principle holds;
- recombination and mutation are exclusive operators when applied to the same individual;
- a new operator called merging is introduced;
- the algorithm stops when, after an a priori fixed number of iterations, no significant change occurs in the population.

The algorithm starts with a large initial population whose size may be reduced at every generation. The sub-populations have an arbitrary structure and they become better separated with each iteration. The stepping stone search mechanism provides the possibility that each individual takes part in the process of forming the new generation. Thus, by making use of it, each individual is considered for mating. Its mate will be determined by applying a local selection scheme. This local interaction principle is a natural thing to do, since from a biological point of view it is more likely that individuals from the same sub-population mate rather than from different ones. If a second chromosome is found within that range, then they will recombine. Else mutation will be applied to the current individual. It is also possible to use a global search mechanism for the mate instead, in some situations, but this is of no interest with respect to the problem at hand. As an observation, selection is carried out both globally - it is the case of the first chromosome — and locally — as seen for the second parent, if there is one. If two chromosomes are very similar, they are merged into a single one, by usually taking their mean.

3. Genetic Chromodynamics Clustering

The clustering method we propose uses GC principles. A standard genetic chromodynamics model is used, together with some data-related features. The data points allow both numerical and nominal attributes. We have tried to develop an algorithm that would work perfectly with any kind of data. And it almost does. The problem is that for a special type of data, e.g. geometrical instances, the evaluation function will not work in most of the cases, because it should be more or less problem dependent.

3.1. **Representation. Initial population.** Each instance from our database represents a chromosome, every attribute of our record being thus a gene. For example if we are dealing with a database for describing the features of different candidates present for some secretary positions, with the structure

(typing skills, number of foreign languages known, years of experience),

then all the chromosomes will have the exact same structure. Let us denote by c the current chromosome. Therefore its value could be for example:

$$c = \textit{((computer - 0.25, typewriter - 0.14), 2, 4).}$$

An important feature of our approach is that each chromosome allows numerical as well as nominal attributes. The problem we were faced with was that no nominal variables can be used in our mathematical computations. Thus we have represented our nominal data as fuzzy.

The initial population is made of the data points.

3.2. **Fitness function.** First of all, we have to define the similarity measure between two chromosomes, since our function is built upon its expression. We have used a weighted similarity measure, since each attribute of our data has a different degree of importance in the field they are extracted from.

$$distance(a, b) = \sum_{k=1}^{n} compare(a_k, b_k),$$

where a and b are the two chromosomes and n represents the number of attributes.

At this point there are two cases. First of all, if we are dealing with numerical attributes, the difference between the two attributes was considered the square weighted Euclidian similarity measure, that is:

$$compare(a_k, b_k) = (a_k - b_k)^2 weight_k,$$

where $weight_k$ is a positive number specifying the importance of attribute k.

In the other case, of the nominal attributes, the formula was considered the max-min distance specific to fuzzy data:

$$compare(a_k, b_k) = \max(\min_{i=1}^{n_k})(a_k^i, 1 - b_k^i) weight_k,$$

where n_k is the number of values for the k-th attribute of the chromosome.

Now the expression of the fitness value is as follows:

$$eval(pop_i) = \sum_{i=1}^{popNo} 1/e^{distance(pop_i, pop_j)},$$

where pop denotes the vector representing the population of chromosomes, $popNo$ meaning the current population size and pop_i meaning the current chromosome.

3.3. **Selection operator.** The mate for the current chromosome, c, will be selected within its pre-determined mating region. This region is defined, mathematically speaking, as the closed ball $V(c, r)$, where we specify the radius, r. Proportional selection will be used from this point on.

3.4. **Variation operators.** Standard recombination and mutation operators are used for guiding the search process. Merging is an additional variation operator.

id	**outlook**	**temperature**	**humidity**	**windy**
x_1	(sunny-0.78,overcast-0.45,rainy-0.20)	(hot-0.90,mild-0.50,cool-0.10)	(high-0.78,normal-0.12)	(true-0.13,false-0.90)
x_2	(sunny-0.80,overcast-0.34,rainy-0.10)	(hot-0.80,mild-0.40,cool-0.20)	(high-0.80,normal-0.20)	(true-0.89,false-0.23)
x_3	(sunny-0.30,overcast-0.85,rainy-0.34)	(hot-0.90,mild-0.30,cool-0.10)	(high-0.90,normal-0.30)	(true-0.16,false-0.77)
x_4	(sunny-0.10,overcast-0.50,rainy-0.90)	(hot-0.40,mild-0.80,cool-0.50)	(high-0.70,normal-0.10)	(true-0.22,false-0.86)
x_5	(sunny-0.13,overcast-0.50,rainy-0.70)	(hot-0.10,mild-0.50,cool-0.80)	(high-0.30,normal-0.80)	(true-0.15,false-0.88)
x_6	(sunny-0.20,overcast-0.40,rainy-0.87)	(hot-0.20,mild-0.40,cool-0.90)	(high-0.30,normal-0.79)	(true-0.77,false-0.30)
x_7	(sunny-0.50,overcast-0.80,rainy-0.30)	(hot-0.30,mild-0.20,cool-0.92)	(high-0.40,normal-0.98)	(true-0.89,false-0.20)
x_8	(sunny-0.90,overcast-0.70,rainy-0.10)	(hot-0.60,mild-0.80,cool-0.20)	(high-0.84,normal-0.22)	(true-0.14,false-0.88)
x_9	(sunny-0.78,overcast-0.34,rainy-0.20)	(hot-0.20,mild-0.60,cool-0.96)	(high-0.13,normal-0.95)	(true-0.10,false-0.98)
x_{10}	(sunny-0.10,overcast-0.50,rainy-0.70)	(hot-0.10,mild-0.90,cool-0.50)	(high-0.24,normal-0.87)	(true-0.34,false-0.68)
x_{11}	(sunny-0.80,overcast-0.30,rainy-0.10)	(hot-0.20,mild-0.87,cool-0.40)	(high-0.32,normal-0.89)	(true-0.56,false-0.45)
x_{12}	(sunny-0.40,overcast-0.90,rainy-0.30)	(hot-0.12,mild-0.90,cool-0.60)	(high-0.82,normal-0.30)	(true-0.85,false-0.30)
x_{13}	(sunny-0.20,overcast-0.90,rainy-0.50)	(hot-0.90,mild-0.50,cool-0.20)	(high-0.40,normal-0.80)	(true-0.65,false-0.22)
x_{14}	(sunny-0.10,overcast-0.30,rainy-0.90)	(hot-0.40,mild-0.78,cool-0.11)	(high-0.98,normal-0.14)	(true-0.94,false-0.12)
	0.2	0.3	0.1	0.4

TABLE 1. The weather data set

mating region	mutation step	merging radius
0.5	0.07	0.4

TABLE 2. Algorithm parameter values

3.4.1. *Merging.* Each chromosome from the current population will be taken into consideration, observing whether there are other chromosomes similar to it behind a certain threshold called merging radius. If that should be the case, the best one from the group will be kept, and the others will be deleted from the population.

3.5. **Stop condition.** The algorithm stops when, after a number of iterations, considered equal in value to the number of the objects in the data set, no new offsprings are accepted in the population.

The last population provides the optimal clustering. Its members correspond to the centers of the resulting clusters and they also hold the information regarding the distribution of the initial data points to these centers.

3.6. **Other parameter settings and experimental results.** Consider a fictional data set that describes the weather conditions for playing some unspecified game [7] given in Table 1.

We consider the values for the other parameters involved given in Table 2.

The corresponding classes are:

$$A_1 = \{x_2, x_{14}, x_{12}, x_4, x_3, x_1, x_8\},$$

$$A_2 = \{x_9, x_5\},$$

$$A_3 = \{x_{10}, x_7, x_6\},$$

$$A_4 = \{x_{11}\}$$

and

$$A_5 = \{x_{13}\}.$$

The run of the corresponding program provides in 9 out of 10 cases, the following grouping:

(i) instances x_6, x_7 in a cluster,
(ii) instances x_5, x_9 in a cluster,
(iii) instances x_2, x_{14} in a cluster, and
(iv) instances x_3, x_4, x_8, x_1 in a cluster.

3.7. **Conclusions and future work.** The initial population size is probably rather small considering only the instances in the data set. A population consisting of the data which is tried to be clustered, maybe with a slight modification in their values, on the one hand, and a double number of randomly generated data points, on the other hand, would probably lead to better results.

References

[1] Dumitrescu, D., Genetic Chromodynamics, Studia Universitatis Babes Bolyai, Ser. Informatica, 2000, 39–50
[2] Dumitrescu, D., A New Type of Evolutionary Metaheuristics, 2003. (to appear)
[3] Dumitrescu, D., Genetic Algorithms and Evolution Strategies, Blue Publishing House, Cluj-Napoca 2000
[4] Dumitrescu, D., Lazzerini, B., Jain, L., C., Dumitrescu, A., Evolutionary Computation, CRC Press, Boca Raton, Florida, 2000
[5] Dumitrescu D., Gorunescu R., Adaptive Prototypes in Evolutionary Clustering, Research Notes in Artificial Intelligence and Digital Communications, 103, 2003, 48–55
[6] Michalewicz, Z., Genetic Algorithms + Data Structures + Evolution Programs, 2nd edition, Springer Verlag, 1992
[7] Witten, I., H., Frank, E., Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, 1999

Faculty of Mathematics and Computer Science, Department of Computer Science, Babes-Bolyai University, 3400 Cluj - Napoca Romania
E-mail address: ddumitr@cs.ubbcluj.ro

Faculty of Mathematics and Computer Science, Department of Computer Science, University of Craiova, 13 Al. I. Cuza 1100 Craiova Romania
E-mail address: ruxandragorunescu@yahoo.com

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

COORDINATION AND REORGANIZATION IN MULTI-AGENTS SYSTEMS, II

ALINA BACIU AND ADINA NAGY

Abstract. A method of considering coordination and reorganization as keys in achieving (organizational) multi-agent system adaptation in unknown situations is proposed. Within a not totally predictable environment multi-agent systems are prone to failures. In such unpredicted situations the system must be able to adapt in order to accomplish its purpose.

The proposed system architecture is a combination of MOISE+ and MOCA concepts. These two models have been presented in Part I.

In this part a new multi-agent system model is proposed. In this proposed model we reconsider the MOCA and MOISE+ notion of role. Our aim is to overcome the main drawbacks of these models. Moreover, the notion of behaviorist role in MOCA is enlarged through several features of roles from the MOISE+ model. We propose some strategies for system's dynamics and coordination that can assure the system adaptation.

Additional keywords: Multiagent Systems Reorganization, Role Endorsement Mechanism

1. Introduction

The goal of the new proposed model is to overcome the main drawbacks of MOCA and MOISE+ models. The aim is two-folded. On the one hand, we suggest a way to achieve the MAS adaptation at environment changes. For this purpose we state some reorganization rules in the management group (endowed with the organizational dynamic, this group idea exists in MOCA) following MOISE+ model of missions. On the other hand, we propose a role endorsement mechanism associated with MOCA-organizational structure and roles, by using the global planning mechanism of MOISE+. This is realized through the integration of the notions of role in MOCA and MOISE+.

2. MAS Adaptation at Environment changes

MAS adaptation is needed because, generally, the system designer cannot predict all the situations the system has to face or because there are some unknown

Received by the editors: October 8, 2003.

2000 *Mathematics Subject Classification.* 68T05.

1998 *CR Categories and Descriptors.* I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *Multiagent systems, Coherence and coordination*.

parameters that define some situations. In unpredictable situations the system must manage in order to achieve its global goal. In what follows we suggest some research directions that would lead to an acceptable system behavior in such unpredictable situations.

The first proposal is related to adaptation in the management group. The second proposal refers to system reorganization coordinated by the agents in the management group.

2.1. **Management group adaptation.** The notion of management group appears also within MOCA platform. The purpose of this group is to manage the organization dynamics meaning group formation, assigning agents to roles, agents entering and leaving a group (Amiguet, 2003). The management group has one agent playing the Yellow Pages role, other agents playing the manager role and agents that wish to enter an organization play the demander role.

The role of Yellow Pages agent is to keep track of what groups are created and to provide this information to agents requiring it. When an agent wants to enter a given group and this group is not created yet, the group will be created and the agent will be assigned the role of manager of this group. From now on any agent that wants to enter or leave the group must communicate with the manager agent.

This group of management could be the entry point of system failures, in case some manager agent fails. In order to find out when a problematic situation has appeared, the agents in the management group have an image about all the others, image that is changed when the agents send messages about their status (Kumar, Cohen, Levesque, 2000).

Following the model of goals and missions from MOISE+ model, the designer of the MAS should specify some rules that would make the agents from the management group to deal with situations in which one or more agents from this group have become unavailable. When some manager agent cannot properly work, the other agents must commit to some specified missions. These missions should impose to available manager agents to take and accomplish the tasks of the agent that became unavailable. They also have to restore somehow the agents with problems. Indeed the whole system performance would decrease but the system failure situation is avoided as long as at least one manager agent is still available.

The proposed solution can only increase system robustness and adaptation in critical situations.

2.2. **System reorganization.** The designer of MAS can imagine some special situations when system reorganization is needed in order to achieve system's global goal. System reorganization means a new organization structure in terms of positions and agents occupying these positions.

A *position* represents *potential roles assignment* within a structure. A position may or may not have an agent assigned to it. At a given moment in time, only one agent can be associated with a given position.

A situation can be identified by some critical parameters. We can tell that a situation is the expected one if some predicates representing that situation are true (of course with some error degree). For some special situations the system designer might know the most appropriate structure for system's best performances.

We saw that MOISE+ model offered the possibility of specifying the minimum and maximum number of positions that can be taken in a group, the minimum and maximum number of subgroups a group can have and also other system parameters. This was a way of defining very flexible organizations. But in some situations it can be known *a priori* which is the best structure that should be used. For example in the soccer game: in a critical situation the system would be more efficient if more agents are in defensive positions.

For such predictable situations the system designer could provide some agent diagrams (Mellouli, Mineau and Pascot, 2002) telling which agents should be put in which positions.

When detecting such situations the management group must start the reorganization operation. In other situations, when it is obvious that the current structure is not the best one but no agent diagram is provided, the management group could decide by itself how reorganization is made.

After the reorganization action, no matter if it was a priori planned or the management group decided it, the taken solution must be given a weight representing its success rate or its current weight must be updated.

This approach of giving feedback for each reorganization action that is made would help the agents next times to adopt a solution that has the greatest success chances.

3. Role Endorsement Mechanism

A strong constraint that is imposed by MOCA platform is that relations between agents in different groups are allowed only if the same agent plays roles in both groups and there are relations between these roles. This constraint is also meet in the proposed model.

First of all we must state two major changes that are made to MOISE+ model:

- relations between agents follow the constraint described above;
- an organization cannot exist without the management group.

The role endorsement mechanism will be exemplified by the following example.

3.1. **An Example for the Structural Dimension in MOISE+ and MOCA.**

Let us consider a MOISE+ organization of soccer players, with its three dimensions: structural, functional and deontic. The structural dimension corresponds to the organizational structure and is presented in Figure 1:

It is obvious that in such a structure, three points of view may be identified: coaching, attack and defense. MOCA allows for the possibility to separate them into three distinct organizational structures: a coaching organization, an attack

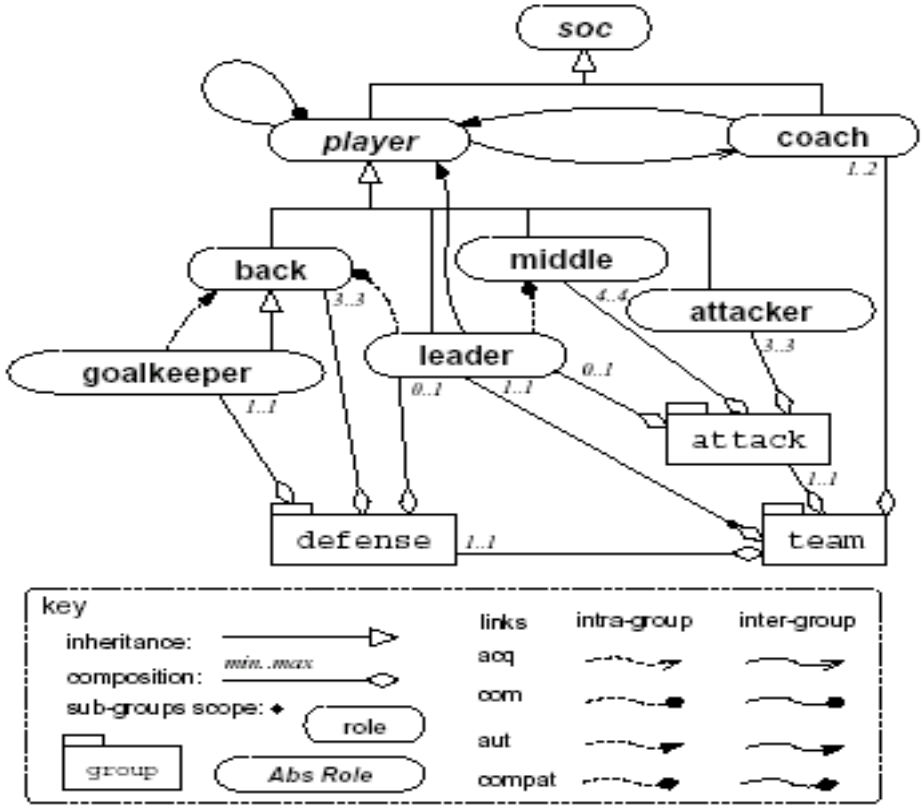

FIGURE 1. Structure of a soccer team

organization and a defense organization depicted in Figure 2. Each adverse team will have to follow such a structure.

The multi-agent system, that is the instantiation of the organizational structure, is initially composed of two instantiations of the Coaching organization (one for each team, with the correct cardinalities for each role) and from an Environment organization, composed from the soccer ground, the ball, the agents in expectation (the reserves) and the coach agent.

The advantage of a multi-view point approach (one view for attack, one for coaching, containing coaches' behavior and basic players' behavior, etc.) on the 'soccer team' is that there is no further need to define any sub-role. Every agent is able to take one or several roles in any organization, according to instantiation rules which need to be specified. No distinction is needed between abstract roles

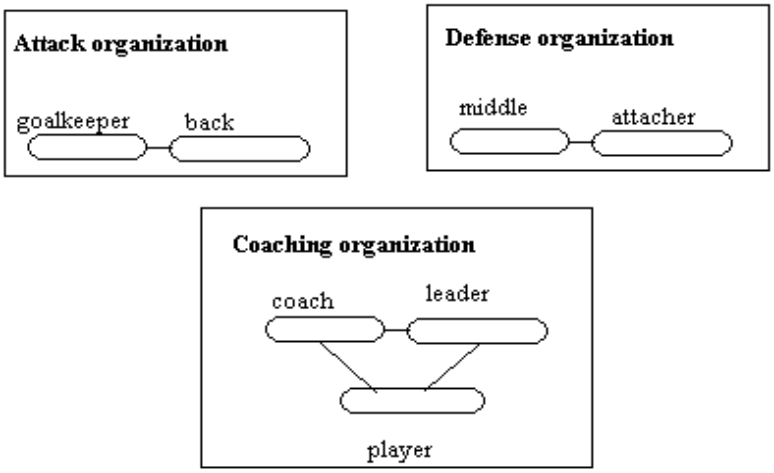

FIGURE 2. Soccer team organizations

and roles, as in MOISE+. Agents will simply be able to take two or several roles in the same time, as 'player' and 'middle' or 'goalkeeper', 'leader' and 'player'. Each of these organizations will be instantiated in one or more groups, which will interact through their shared agents.

MOCA furnishes to any agent a very basic set of skills, related to its ability to communicate, endorse and leave an organization (endorse and leave competences). Acquaintance relations between roles, minimum and maximum cardinalities for roles instantiations are specified, as well as the relations between roles, but a finer cardinality, which might be 'situation-dependent', is desirable. We propose to do this by introducing the notion of *position* of an agent and through MOISE+ like (deontic) predicates.

3.2. **An Example of Behaviorist Role.** There are only a few examples of MOCA organizations. We will exemplify the way MOCA roles are given content. The behavior of every role is fixed, and thus roles represent norms on agents' behavior.

Let us define a *player* role description through a state chart (Amiguet, 2003) as in Figure 3.

The three default states B, D and F concern the knowledge about the players and ball locations and the ability to receive indications from teammates or coach. A1 and A2 are *or-states*, while A and F are *and-states*, which are executed in parallel. The state H allows propagating indication to other roles (ex. 'middle', 'goalkeeper') endorsed by the same agent.

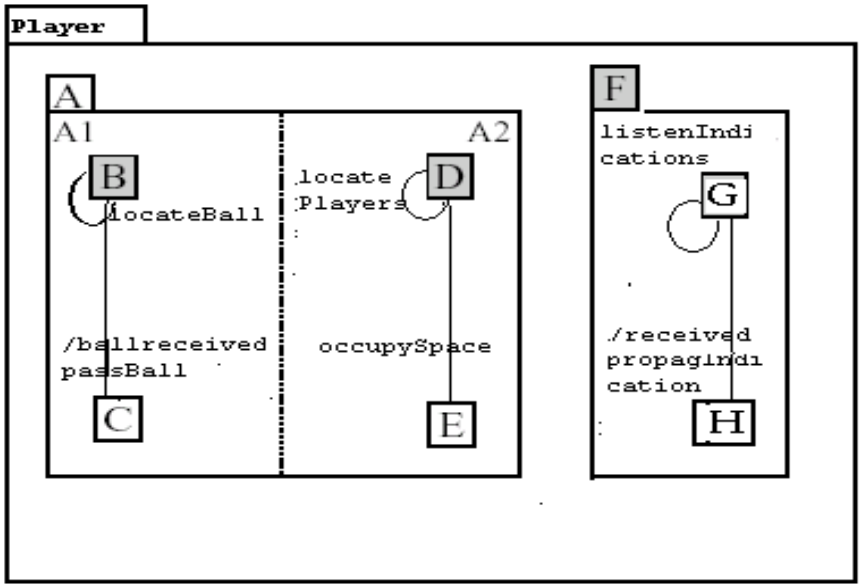

FIGURE 3. Player role

3.3. **Agent Position.** Assume every individual agent in the team (excepting the coaches) has endorsed a player role. We are now able to use the individual goals graph built through MOISE+ goal decomposition.

We will first define the *status* (social status) of an agent. This (sociology inspired) notion defines *the set of roles an agent endorses at a given moment.*

The notion of *position,* defined before can, then, be given a more precise meaning. The position is the *potential status of an agent.* It represents *all the roles and competences an agent might have in a given moment.* The position will be computed from agent's competences, the roles it already endorses, the relations between roles, organizational constraints regarding the roles compatibilities and agent's individual goals.

The *position* of every team member will be equally composed by a set of basic competences like - for the example of soccer agents - being able to *move* within the ground, to *perceive* the locations of other players, to *receive* indications from them and from the coach. Some of these skills are provided by the **player** role exemplified above. Further skills will be provided by the other roles. Individual positions of the soccer player agents are then defined by the roles already endorsed

and other conditions expressed through deontic predicates. For example, an agent who is already a coach may be middle if he is not retired from the team; an agent who is a goalkeeper cannot take the role of attacker.

3.4. **Goal Driven Mechanism for Role Endorsement.** Once the positions of any agent are computed, the functional dimension of MOISE+ expressed through goal description and potential missions (that is succession of goals) can be used.

The MOISE+ functional decomposition is exemplified bellow in Figure 4.

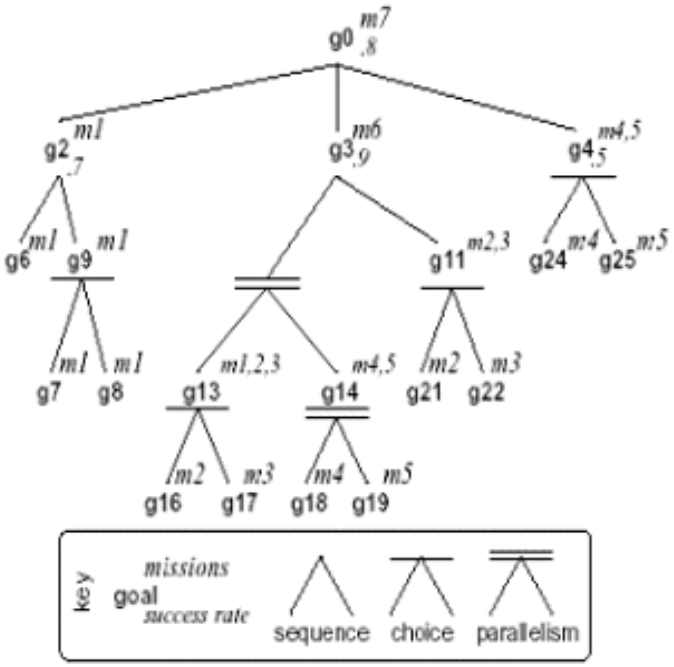

FIGURE 4. Soccer team social scheme to score a goal

We are now able to show how role allocation (endorsement and leaving) can be driven by individual goals. We consider two situations when role allocation is driven by agent individual goal:

(i) Suppose that g24 was already reached. In most of these cases, according to the ball's position (considered as input from the environment) and to an established defense strategy (not exemplified here) ***left attacker*** agent might wish to take a role in the defense organization;

(ii) Suppose now that the ball is in the middle field and a ***middle*** agent may choose the mission composed of the goals {g2, g9, g13, g18, g25}. Then, it will have to endorse the role of ***attacker*** for this individual action.

Goal	Goal description
g0	score a soccer-goal
g1	the ball is in the middle field
g2	the ball is in the attack field
g3	the ball was kicked to the opponent's goal
g4	a teammate has the ball in the defense field
g6	the ball was passed to a left middle
g7	the ball was passed to a right middle
g9	the ball was passed to a middle
g11	a middle passed the ball to an attacker
g13	a middle has the ball
g14	the attacker is in good position
g16	a left middle has the ball
g17	a right middle has the ball
g18	a left attacker is in good position
g19	a right attacker is in good position
g21	a left middle passed the ball to a left attacker
g22	a right middle passed the ball to a right attacker
g24	a left attacker kicked the ball to the opponent's goal
g25	a right attacker kicked the ball to the opponent's goal

TABLE 1. Semantics for notations from Figure 4

MOCA allows a dynamic role allocation, but the mechanism is entirely left to the designer. We have seen now that the allocation may be done thanks to dynamical individual goals agents may have. The identification of individual goals to goals from a given strategy might be done in several ways: through the individual reasoning about a situation, through indications received from the leader or the coach, or through the past experience agent has recalled.

Thus, we have shown how the purpose of using a MOISE+ like functional decomposition to drive the MOCA role allocation can be reached. The formalism behind this example will be described in a further paper.

4. Group dynamics

It is difficult and often infeasible to specify Multi-Agent-Systems completely in advance, because there are frequently unforeseen situations that agents may encounter.

The MOCA platform enhances agents with the capability of entering and leaving a group. These operations are managed by the manager agent who is the first agent that required entering the group before the group has been created. After the group creation all agents that want to enter or leave the group must ask this to the.

In this context we can see the role having a double interpretation: it reflects the competencies that an agent has and that the agent should provide to the other agents.

The manager agent should be endowed with a reasoning capacity in order to allow an agent to enter the group or not. The acceptance of an agent by a group depends on the fact that the utility of the group increases. The manager agent disposes a mechanism of punishment and favoring to make agents agree on roles. On the other hand, an agent joins only a group if its own utility increases, too.

We consider a group as being formed from agents having common interests. An interest can be the desire or need to share resources and competencies. Within an organization agents have different roles reflecting the competencies necessary for accomplishing the common goal. This competence based approach comprises deliberative agents which are aware of the roles they are playing and of those they want to play.

So we can define a group as the set of roles which identify the positions which individual agents can play. As defined in MOCA the role is defined by the competences associated to it. Each individual agent is able to play different roles in a society depending on its individual competencies.

An agent has a set of desired roles that he wants to play. Instead the group has a set of expected roles that agents entering the group should play. We define the committed role as the role the agent has committed to play within the group.

When an agent enters a group something like a convention is created between the agent and the entered group. In fact, a convention is something like a social norm which allows agents to increase the utility of an organization and also their own one. In fact a convention describes the structure of the group that has been statically defined by the system designer (the maximum and minimum number of agent playing a certain role) or a new convention could have been created in a system reorganization phase (see System reorganization).

How do agents now agree on a convention to form an organization? We need something to express the motivation of a group to accept/refuse an agent and the motivation of an agent to accept/refuse a role. A very useful mean is the definition of utility functions.

The utility function for an agent depends on the roles it wants, the roles it has already committed to. Also the organization has a utility function. This function depends mostly on its convention. Examples of utility function definitions can be found in (Glasser and Morignot, 1997).

This dynamic behavior of an organization permits the group evolution over time. Autonomous agents are required to be able to modify their local knowledge in such a way that they can agree with other agents to build a group.

Thus when a candidate agent having reactive, cognitive, cooperative and social competencies applies for a group membership, the group manager will favor the agent's desired roles that augment the organization utility function.

This management of group dynamic makes it possible the organization to draw benefits from its new members and to be able to answer the environmental changes.

5. Conclusions

This paper represents a starting point in combining the two organization centered models, namely MOCA and MOISE+. The notion of *status* of an agent and the notion of *position* are defined. The notion of position is combined with the decomposition of global goals in *goal schema* and *missions*, in order to direct the MOCA-role assignment strategies and role endorsement through a dynamic choice of individual goals.

References

[1] Amiguet, M., Müller, J-P., Báez, J, Nagy, A. 2002. The MOCA Platform: Simulating the Dynamics of Social Networks, Workshop of Multi-Agent–based simulation, MABS'02, Barcelona
[2] Amiguet, M. 2003. MOCA: un modèle componentiel dynamique pour les systèmes multi-agents organisationnels, thèse de doctorat, Université de Neuchâtel (Switzerland)
[3] Baez, J. 2002. Extension et consolidation de la plate-forme organisationnelle MOCA, mémoire de diplôme, Université de Neuchâtel (Switzerland)
[4] Barbuceanu, M., Lo, Wai-Kai. 2000. Integrating individual and social reasoning models for organizational agents
[5] Ferber, J. 1999. Multi-Agent Systems, Addison Wesley
[6] Hannoun, M., Boissier, O., Sichman, J.S., Sayettat, C..1999. Moise : Un modèle organisationnel pour la conception de systèmes multi-agents, Editions Hermés
[7] Hübner, J.F., Sichman J.S., Boissier O. 2002 A Model for the Structural, Functional, and Deontic Specification of Organizations in Multiagent Systems
[8] Olivier Gutknecht, O., Ferber, J. 2002. MadKit v2.0.1, LIRMM, Université Montpellier II, 2000, http://www.madkit.org
[9] Parunak, H., Odell, J. 2001.Representing social structures in UML, AOSE2001
[10] Odell, J., Parunak, H., Fleischer, M. 2003. The Role of Roles in Designing Effective Agent Organizations, *Software Engineering for Large-Scale Multi-Agent Systems*, Alessandro Garcia et al, LNCS, Springer
[11] Kumar, S., Cohen, P. R., Levesque, H. 2000. The Adaptive Agent Architecture: Achieving Fault-Tolerance Using Persistent Broker Teams, Fourth International Conference on Multi-Agent Systems (ICMAS-2000), Boston
[12] Mellouli, S., Mineau, G., Pascot, D. 2002. Multi-Agent System Design, ESAW'02, Engineering Societies in the Agents World, Madrid, Spain
[13] Nagy, A. 2002. Behaviorist organizational models for the MAS – a state of the art, Technical Report, University of Neuchâtel, Switzerland
[14] Glasser, N., Morignot, P. 1997. The Reorganization of Societies of Autonomous Agents

Faculty of Mathematics and Computer Science, Babes-Bolyai University of Cluj-Napoca, Romania
E-mail address: alina_baciu@yahoo.com

Institute of Computer Science and Artificial Intelligence, University of Neuchâtel, Switzerland
E-mail address: adina.nagy@unine.ch

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

RSDNET: A WEB-BASED COLLABORATIVE FRAMEWORK FOR BUILDING MULTILINGUAL SEMANTIC NETWORKS

NATHANIEL AYEWAH, RADA MIHALCEA, VIVI NĂSTASE, AND DOINA TĂTAR

Abstract. We present a system (RSDnet) that allows non-expert Web users to contribute towards building a multilingual lexical resource. Our study focuses on the Romanian-English language pair, and the target resource is a Romanian WordNet strongly connected to the English WordNet. We use a bilingual dictionary, a monolingual definition dictionary and documents on the Web to build synsets, attach them a gloss, and provide some examples. The results of our semi-automatic acquisition system are judged by two human judges, and they are compared to automatic approaches to building a Romanian WordNet.

Keywords: Semantic dictionary, RDSnet

1. Introduction

In order to obtain a system that provides expertise in a specific domain, the knowledge of that domain must be made available in a format that the system can use. Developers of software often do not have the knowledge of such specific domains, and experts in the field do not have the knowledge to create such a knowledge base. This has led to a new trend, in which software developers write tools that allow experts to readily formalize their knowledge through the system provided, which then encodes this input in a format that a system can use [7].

Language is a field that all people are experts in. We offer them RSDnet – a tool freely available on the Internet, with a friendly interface, through which Web contributors participate in the construction of a multilingual semantic network, by validating automatically suggested synonym sets.

We present in this paper the paradigm behind this system, the implementation and the interface, the role of the user, and an analysis of the results obtained so

Received by the editors: March 20, 2004.

2000 *Mathematics Subject Classification.* 68T50, 68N99.

1998 *CR Categories and Descriptors.* I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *Language models*.

far. The results gathered were analyzed by two human judges, and compared to results obtained in other similar endeavors.

2. Related Projects

The idea of harnessing the knowledge of experts in a particular field in order to gather data has found many applications.

The Rapid Knowledge Formation project [7] is geared towards providing experts in various fields with tools that allow them to encode their knowledge in an intuitive way, without needing to acquire programming skills. This is realized by using a graphical interface, which the experts manipulate to form and link concepts [4].

Collecting data over the Web for a variety of AI applications is a relatively new approach. The basic idea behind the broad *Open Mind* initiative [16] is to use the information and knowledge obtainable from millions of Web users to create more intelligent applications. *Open Mind* projects include our own effort – *Open Mind Word Expert* [3] – to build lexically annotated corpora through volunteer contributions. They also include *Open Mind 1001 Questions* [2], which acquires knowledge and *Open Mind Common Sense* [15], a system that collects common sense statements from Web users.

The domain of expertise our project is focused on is language. All native speakers of a language are expert users of their mother tongue. A structured system can help them focus on particular aspects, and harness their knowledge towards the construction of interesting resources. *OpenMind Word Expert* provides such a system, allowing people all over the world to contribute towards building a corpus annotated with semantic information [3].

WordNet [13] is a lexical resource that is used frequently in the NLP community for word-sense disambiguation, question answering and summarization, and other tasks. It's success has led to projects aimed at building equivalent resources for other languages.

[19] show the process of building a multilingual resource based on WordNet 1.5. An inter-lingual index (ILI) provides the connection among WordNet and all the other resources in various languages that are being built. For each language, a core WordNet is manually built for a set of common base concepts. These sets are then enriched with semantic links, and they are expanded in a top-down manner [20]. ILI is however strongly connected to the original WordNet 1.5 resource, making it difficult to port the multilingual network to new WordNet versions. Moreover, ILI does not have the capability of storing word-to-word relations within a synset, and therefore the use of this resource for multilingual applications (e.g. machine translation, cross language information retrieval) is not always straighforward.

[8] show a way of building a semantic network using a monolingual dictionary, and then merging this structure with WordNet, in order to enhance it with the semantic links that WordNet provides.

The *Euro WordNet* project covers languages from western and central Europe (French, German, Italian, Spanish, etc.). *BalkaNet* is a similar project, focused on languages from eastern Europe (Romanian, Bulgarian, etc.). As opposed to the *Euro WordNet* endeavor which emphasized the multilingual nature of the project, *BalkaNet* allows the projects for each language to develop on their own.

[14] propose an automatic way of building candidate synsets in the target language (Bulgarian) using *WordNet*, an English-Bulgarian dictionary and a Bulgarian-English dictionary. The candidate synsets (called e-sets) are built by translating each English word in a synset into Bulgarian using the English-Bulgarian dictionary, and then choosing from the possible senses of the word by cross-referencing the results using the Bul-garian-English dictionary. A function is used to evaluate the goodness of the e-sets. Ultimately, a linguist chooses from the proposed candidates. The algorithm proposed was found to work well with nouns.

[10] use a similar process as [14] to build a *Romanian WordNet*. The algorithm they employ covers nouns, adjectives and verbs. Again, two bilingual dictionaries are used to translate words in synsets, and perform word-sense disambiguation between possible senses. The system developed is language independent, and free for tryouts [18], [12]. We use this system to test the results of our acquisition experiments.

[1] propose a semi-automatic approach to building ItalWordNet, in which a system uses the English WordNet and a bilingual dictionary to propose a linguist supervisor possible synsets. The user can also input language-specific synsets through a special interface.

3. Resources

One of the most important objectives targeted by the RSDnet system design is to facilitate the task of the non-expert contributor as much as possible. That is, rather than asking the user to look for external resources for word translations, definitions, and examples, we try to provide several such resources directly on the system Web site. With such resources linked directly from the RSDnet page, the task of the users is greatly simplified – they select the right information from a pool of readily available information and usually do not have to seek additional resources.

3.1. **WordNet.** WordNet is the primary information source that we use in RSDNET for the construction of a new semantic network. WordNet is a Machine Readable Dictionary developed at Princeton University by a group led by George Miller [13], [9].

WordNet covers the vast majority of nouns, verbs, adjectives and adverbs from the English language. The words in WordNet are organized in synonym sets, called *synsets*. Each synset represents a concept. WordNet 1.7 is the latest WordNet version and it was released in July 2001. It has a large network of 144,680 words, organized in 109,373 synonym sets, called *synsets*. Table 3.1 shows the number of nouns, verbs, adjectives and adverbs defined in WordNet 1.7, and the number of synsets for each of these parts of speech.

Part of speech	Words	Synsets
Noun	107,929	74.487
Verb	10,805	12,753
Adjective	21,364	18,522
Adverb	4,582	3,611
TOTAL	144,680	109,373

TABLE 1. Words and synsets in WordNet 1.7

WordNet also includes an impressive number of semantic relations defined across concepts (249,425 relations in WordNet 1.7). For instance, the following relations are explicitly encoded in WordNet:

- Hypernymy/hyponymy relation (IS-A), as in *tree* IS-A *plant*.
- Meronymy/holonymy relation (HAS-A), e.g. *car* HAS-PART *airbag*.
- Antonymy, defined for all parts of speech, e.g. *beautiful* (vs.) *ugly*.
- Entailment, which is a pointer defined only for verbs, as *limp* entails *walk*.
- Pertainimy, involves adjectives, adverbs and nouns, and groups together words that are related, as *parental* pertains to *parent*.

Note that semantic relations are defined among *concepts*, and not among *words*, and therefore the belief is that the same semantic relations hold in any language, independent of the *words* that are used to lexicalize a given *concept*. The goal of RSDNET is to identify, with the help of Web users, these *concept* lexicalizations specific to a given language (e.g. Romanian), and build a resource similar in structure to the original English *WordNet* in a much shorter period of time than if starting from "scratch".

3.2. **Bilingual dictionaries.** RSDNET uses bilingual dictionaries to suggest translations for a given English word in a WordNet synset. We use a combination of several dictionaries that were identified online. Currently, RSDNET uses an English-Romanian dictionary with about 75,000 entries, out of which about 40,000 are word-to-word translations, and the rest represent phrasal translations. This dictionary is used to suggest candidate translations in Phase 1 in the Web interface, as described in section 4.

3.3. **Monolingual Dictionaries.** Once a synset have been selected (we point out that at a time only one synset of a word is selected) , RSDNET attempts to suggests definitions and examples for all the words in the synset. To this end, we are using a monolingual Romanian dictionary, consisting of about 35,000 definitions for the most frequent words in the Romanian vocabulary. In future versions of RSDNET, we plan to use an augmented monolingual dictionary, by integrating the output of "DEX online," a collaborative effort for building an online alphabetic Romanian dictionary initiated by Cătălin Frâncu[1].

3.4. **Romanian Corpus.** RSDNET makes several suggestions for synset word examples, to complete the synset gloss. Examples are extracted from a 400 million words corpus, consisting of a collection of Romanian newspapers collected on the Web over a three years period (1999-2002). Alternatively, RSDNET users can use search engines to directly identify examples on the Web. The RSDNET interface includes links to several search engines (currently, we link to Google, AltaVista, Lycos), and search queries are automatically formed with the synset words, for increased efficiency. Moreover, users can complete their own short examples.

4. Web Interface

The strength and lure of Web data collection systems is the seemingly limitless availability of users that possess the knowledge the system aims to acquire. The RSDNET Web interface aims to maximize the benefit received from this resource by providing facilities to simplify the data collection process for the user, increase the contributions by each user and minimize the occurrence of errors. The interface simplifies the data collection process by dividing it into phases and providing suggestions whenever the user needs to provide input to the system. It also uses a scoring system to reward users with recognition and prizes for significant contributions. An administrative facility allows an exclusive group of experts to review the inputs and make corrections where necessary.

[1]http://dex.francu.com

4.1. **The Phases.** The RSDNET interface uses four phases to guide the user to their final destination of capturing a Romanian synset. These phases are transparent to the user and allow him or her to focus on a small part of the problem. Briefly, these phases allow the user to:

- choose a synset to define,
- find appropriate lexicalizations of the word or words in synset,
- develop or retrieve definitions and sample sentences that match the word (words) in synset , and
- review the inputs to eliminate errors.

The preliminary phase, *Phase 0*, displays a list of random English synsets from WordNet. Beside the set of words, the system also displays the synset's gloss from WordNet, which describes its meaning.

For example, RSDNET may display in *Phase 0* a synset with the words *diversion, deviation, digression, deflection, deflexion* and the gloss *turning aside (of your course or attention or concern): "a diversion from the main highway"; "a digression into irrelevant details"; "a deflection from his goal"*.

When running this system, it quickly becomes obvious that some of the synsets in WordNet cover concepts that are not familiar to all users. So this phase includes a facility that allows the user to request another random list of synsets for her to choose from.

Once a synset is selected, and the user constructs the equivalent Romanian synset, the synset is removed from the pool of "available" synsets and moved into a different set containing synsets to be validated.

Phase 1 directs the user to specify the Romanian words that belong in the chosen synset either by translating the words in the English synset or by providing words that are not direct translations of any of the English words. To speed up the process, RSDNET uses an internal English to Romanian dictionary (Section 3.2) to translate the English words. These are ONLY suggestions for the user because the system cannot determine if the translations are correct in the context of the synset. For example, RSDNET translates the word *plant* as *plantă* which is correct if the synset refers to *living organism*, but not if the synset refers to *industrial plant*. It is the role of the contributor to decide on the right translation for a given synset word.

In the example described above for Phase 0, RSDNET correctly suggests the words *deviere, deviere, digresiune, abatere* respectively for the first four words.

Figure 1 shows a screen shot of *Phase 1*. As an added benefit of this phase, the translations validated by the user create relationships between English and

Romanian words within the context of the synset. These relationships are stored in a database and could eventually lead to a semantic English-Romanian dictionary.

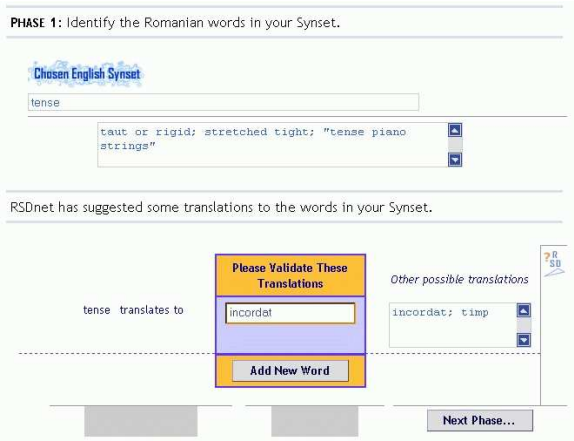

FIGURE 1. The top section of *Phase 1* keeps the English synset in view while the bottom section directs the user to make changes to Romanian translations if necessary.

In *Phase 2*, RSDNET uses an internal Romanian Dictionary (Section 3.3) and a textual corpus (Section 3.4) to suggest definitions and samples respectively. As in *Phase 1*, these suggestions may be out of context and the user needs to validate them or add new entries. *Phase 2* also provides a facility that allows the user to use popular search engines on the Web to retrieve sample sentences.

In the earlier *"diversion"* example, RSDNET does not suggest any definitions, so the user enters one: *o schimbare (în atentie, a drumului, etc.)*. The system provides some samples from which the user selects *o deviere a drumului*, *o digresiune de la subiect*, and *o abatere de la calea cea dreaptă*.

Phase 3 directs the user to review the synset he or she has created to look for errors. Additionally, since Phases 1 and 2 allow the contributor to use simple html markup and to represent special characters using html, this phase gives a visual confirmation that the right markup has been used. From this phase, the user can finally submit his or her contribution to the RSDNET database.

In each of these phases, an online help system provides instructions for the non-expert user. This system also provides a reference for escape sequences that can be used to represent special Romanian characters. For example, the sequence '`\a`' is used to represent ă.

4.2. **The Administrative System.** A major concern with Web based data collection is the introduction of errors into the database because of a user's oversight, malicious intent or limited knowledge of the language. A color-coded administrative Web page was designed to allow select individuals to review and validate the entries into RSDNET, correcting or deleting them if necessary. Figure 2 shows a screenshot of this page, which also shows the administrator the original English synset and the relationships that have been created between English and Romanian words. A field in the database indicates which synsets have been validated.

4.3. **The Scoring System.** RSDNET, like similar projects at teach-computers.org, uses a rewards program to motivate users to make more contributions. When the user submits his or her synset in *Phase 3*, a score is computed based on the number of words in the synset. This is a very simple measure and can be thought of as a measure of the size of the synset. Future scoring schemes may consider the number of definitions and samples provided and penalize the user for incorrect entries. To attract new users and increase retention, we are giving away prizes, on a weekly or monthly basis.

4.4. **Other Interface Features.** Only users that are registered with RSDNET can improve their scores and win prizes. RSDNET provides a simple interface for registering with the system and updating personal information such as an email address. The RSDNET interface also solicits feedback from contributors to look for ways to improve the system.

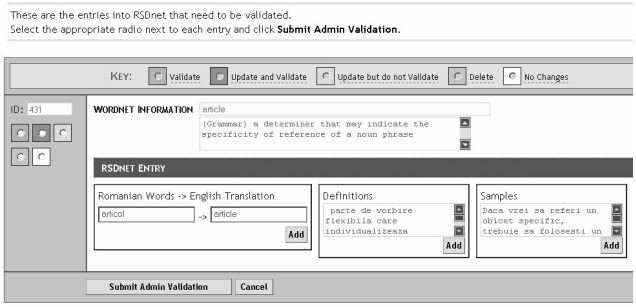

FIGURE 2. On the *Administrative Screen*, an expert can use the radio buttons on the left to make changes to the synset or remove it from RSDnet. The expert can also delete a field in the synset by leaving it blank.

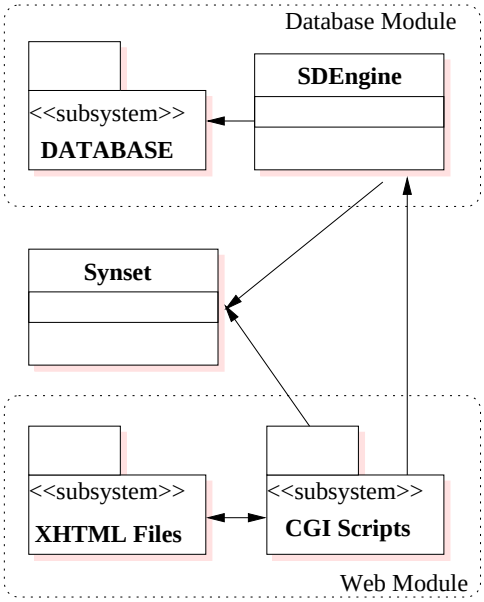

FIGURE 3. The High Level Design divides the system into two distinct modules which use a *Synset* object to encapsulate the information passed between them. *SDEngine* is a Perl object that contains all the queries used to access the database.

5. Designing RSDnet

Figure 3 shows RSDnet's high level design which breaks the system into two modules: a Web module responsible for rendering and manipulating the RSDnet interface and a database module which controls all access to RSDnet's content. The one-directional arrow between the two modules indicates a 'client-server' relationship. The Web module (client) sends requests to the database module (server) to get information from the database. This design, along with a complete specification of the interface between the two modules, made it possible to develop and customize both modules simultaneously and relatively independently. The design aims to make it straightforward for other projects to interface with the modules.

The development of the RSDnet Web module focused on making it functional and extensible. This module is only a prototype and so, for example, does not provide secure login facilities to users adding entries to RSDnet. It does aim to be relatively fast and uses CGI scripts written in **Perl** which is ideal for rapid development and efficient at processing strings. To render the Web interface, this

module uses **xhtml** files as templates which the CGI scripts populate with information from the database module. The look and feel of the interface is achieved using **xhtml** [6], cascading style sheets [5] and JavaScript.

The database module represents the more enduring aspect of RSDNET because it is more likely to be used in other projects especially when RSDNET becomes more comprehensive. It aims to be secure, reliable and well organized. A single **Perl** object, *SDEngine*, provides secure access to the database and contains all the queries that allow the Web interface to manipulate RSDNET's information. The database is designed as shown in Figure 4 to ensure that RSDNET can easily be used to perform many tasks including identifying Romanian synonyms (as a semantic dictionary), retrieving definitions for Romanian words (as a Romanian Dictionary) and providing English translations for Romanian words and vice versa (as a bilingual dictionary).

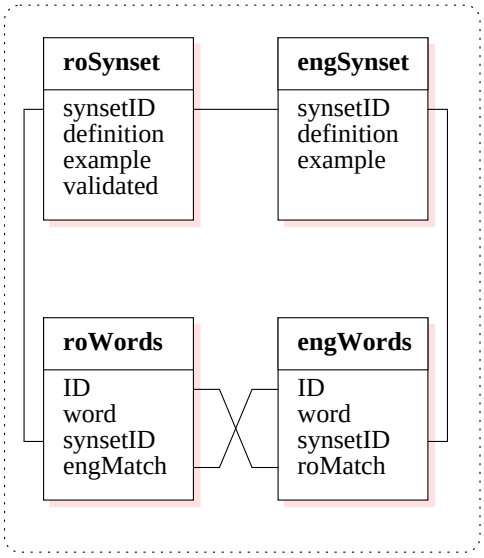

FIGURE 4. The Database maintains a one-to-many relationship between each concept (synset) and the lexicalizations (words) of the concept. At the same time, it maintains the relationships between English words and Romanian words within the context of a synset. A word can occur several times in each of the 'Words' tables if it belongs to more than one synset.

	RSDnet			
	n	v	a	r
Correct synsets	96.6% (57)	100% (13)	92.3% (24)	100% (3)
Partially correct	3.3% (2)	0	3.8% (1)	0
Erroneous	0	0	3.8% (1)	0
Missing	0	0	0	0
Total	59	13	26	3
	GenSynsets			
	n	v	a	r
Correct synsets	18.5% (10) / 63% (34)	–	20.8% (5) / 71% (17)	–
Partially correct	1.8% (1) / 1.8% (1)	–	0 / 0	–
Erroneous	3.8% (1) / 3.8% (1)	–	4.1% (1) / 4.1% (1)	–
Missing	77.7% (42) / 33% (18)	–	75% (18) / 25% (6)	–
Total	54	–	24	–

TABLE 2. Results obtained with RSDnet for noun (n), verb (v), adjective (a) and adverb (r) synsets

6. Evaluation

We have compared the quality of the data obtained using RSDnet with data obtained from a system designed to automatically build a Romanian WordNet [10].

The comparison with automatically obtained data using GenSynsets [18] has led to a few observations. The fact that the ultimate judge in entering data is a human bypasses most errors introduced by the lexical resources we use (the bilingual and the monolingual dictionaries). If for a certain word the dictionary does not provide a translation, the user can enter one himself. In the automatic approach, the system will produce no results for that particular synset, simply because the resources it has available are far from perfect. This is reflected in the difference between the accuracy numbers shown for GenSynsets in Table 2. The second set of results show a different run of the system when the dictionaries that the system used were manually edited to correct spelling and formatting errors. Also, GenSysets processes nouns, verbs and adjectives separately, and expects the dictionaries to provide separate entries for each of these parts of speech. We have used in the comparison the same dictionary that RSDnet uses, which does not have part of speech information to allow us to separate the dictionary entries. Because of these issues, the automatic system produces more erroneous or has more missing synsets than it would with the appropriate dictionaries.

Table 2 shows the comparative results of RSDNET and the GenSynsets system, as evaluated by a human judge. 100 experimental synsets built using RSDNET have been manually validated by two human judges. RSDNET uses WordNet 1.7 as a reference, while GenSynsets was built to work with WordNet 1.6. A program automatically extracts the synsets in the 1.6 version of WordNet that correspond to the synsets translated using RSDNET. Some pairings between the two versions could not be made, and from the 100 synsets we have found 92 in the 1.6 version of WordNet. GenSynsets will work with these. Also, GenSynsets generates synsets only for adjectives and nouns, although theoretically the system also works for verbs [11].

7. What's Next

By constructing RSDNET, we choose a middle way between an automatic system, and a fully manual endeavour of building a semantic network of concepts. The pitfalls of the automatic approach come from the fact that it relies completely on imperfect lexical resources (namely dictionaries), which have a negative impact on the final results, as we have shown in section 6. The other extreme, a manual approach, is expensive in terms of time and human resources. We plan to compare the results of our semi-automatic acquisition with synsets created manually by Romanian linguists [17]. If the quality of our collection fares well in comparison with the one created by specialists (which is very likely, given the fact that the human judges validated the synsets collected until now with minor modifications, as was shown in table 2), RSDNET will be proven to be a worthwhile endeavour.

Although RSDNET provides an environment for building a semantic network for Romanian based on a similar resource for English, the paradigm behind the system is generic enough to be applied for any pair of languages. The requirements are a resource for the original language to be modelled in the target language, and a bilingual and monolingual dictionaries for the target language. The existence of a corpus for extracting samples of usage would also be useful, but not indispensable.

The data collected in RSDNET does not consist only of synsets, but also of word pairs. The system keeps track of the English word and its Romanian translation, in the context of the synset to which the words belong. Such word-to-word translations could prove to be very useful in machine translation, cross language information retrieval, and other multilingual applications, since they show the lexicalization of a specific concept in Romanian and English.

As RSDNET system grows, researchers will want to integrate it in their applications. More interfaces, similar to the interfaces to WordNet, will be needed to

provide access to the validated contents of RSDNET. Downloadable releases, including manuals, will also be needed. Right now RSDNET only provides a human interface. But as Web Services become more popular and other similar projects grow, more machine-centric interfaces will be needed to facilitate collaboration between the different systems.

8. Acknowledgments

The authors would like to acknowledge the contribution of Ryan Farmer, Grace Kim, Elizabeth McLendon and Dr. Donald Evans from Southern Methodist University who helped develop this system.

References

[1] L. Bentivogli, E. Pianta, and C. Girardi. MultiWordNet: Developing an aligned multilingual database. In *Proceedings of the First International Conference on Global WordNet*, Myore, India, 2002.

[2] T. Chklovski. *Using Analogy to Acquire Commonsense Knowledge from Human Contributors*. PhD thesis, MIT, 2003.

[3] T. Chklovski and R. Mihalcea. Building a sense tagged corpus with Open Mind Word Expert. In *Proceedings of the Workshop on "Word Sense Disambiguation: Recent Successes and Future Directions", ACL 2002*, Philadelphia, July 2002.

[4] P. Clark, J. Thompson, K. Barker, B. Porter, V. Chaudhri, A. Rodriguez, J. Thomere, S. Mishra, Y. Gil, P. Hayes, and T. Reichherzer. Knowledge entry as the graphical assembly of components: The SHAKEN system. In *Proceedings of K-CAP 2001*, Victoria, BC, Canada, 2001.

[5] W. W. W. Consortium. Cascading style sheets, 2003. http://www.w3.org/Style/CSS/.

[6] W. W. W. Consortium. Hypertext markup language (html), 2003. http://www.w3.org/MarkUp/.

[7] DARPA. The rapid knowledge formation project, 2000. http://reliant.teknowledge.com/RKF/.

[8] X. Farreres, G. Rigau, and H. Rodriguez. Using WordNet for Building WordNets. In *Proceedings of the COLING-ACL 98 workshop on the Usage of WordNet in Natural Language Processing Systems*, Montreal, Canada, 1998.

[9] C. Fellbaum. *WordNet, An Electronic Lexical Database*. The MIT Press, 1998.

[10] F. Hristea. On the semiautomatic generation of WordNet type synsets and cluster. *Journal of Universal Computer Science*, 8(12):1047 – 1064, 2002.

[11] F. Hristea. On the semiautomatic generation of verb synsets in languages other than English. *Anals of the University of Bucharest*, ANO LII:75–86, 2003.

[12] F. Hristea. On the semiautomatic generation of WordNet type synsets and clusters with special reference to Romanian. *Building Awareness in Language Technology*, pages 113–140, 2003.

[13] G. Miller. Wordnet: A lexical database. *Communication of the ACM*, 38(11):39–41, 1995.

[14] T. Nikolov and K. Petrova. Towards buildling Bulgarian WordNet. In *Proceedings of RANLP 2001*, pages 199–203, Tsigov Czark, Bulgaria, 2001.

[15] P. Singh. The public acquisition of commonsense knowledge. In *Proceedings of AAAI Spring Symposium: Acquiring (and Using) Linguistic (and World) Knowledge for Information Access.*, Palo Alto, CA, 2002. AAAI.

[16] D. Stork. The Open Mind initiative. *IEEE Expert Systems and Their Applications*, 14(3):19–20, 1999.

[17] D. Tufis and D. Cristea. Methodological issues in building the Romanian WordNet and consistency checks in BalkaNet. In *Proceedings of LREC2002 Workshop on Wordnet Structures and Standardisation*, pages 35–41, Las Palmas, Spain, May 2002.

[18] G. D. Ungureanu, F. Hristea, and M. Popescu. GenSynsets, 2002. http://phobos.cs.unibuc.ro/roric/gensynsets.html.

[19] P. Vossen. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers, Dordrecht, 1998.

[20] P. Vossen, L. Bloksma, H. Rodriguez, S. Climent, N. Calzolari, A. Roventini, F. Bertagna, A. Alonge, and W. Peters. The EuroWordNet Base Concepts and Top Ontolgy, 1998. Deliverable D017D034D036 EuroWordNet LE2-4003.

Southern Methodist University
E-mail address: ayewah@engr.smu.edu

University of North Texas
E-mail address: rada@cs.unt.edu

University of Ottawa
E-mail address: vnastase@site.uottawa.ca

Babeş-Bolyai University
E-mail address: dtatar@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

OPTIMAL CLASS FRAGMENTATION ORDERING IN OBJECT ORIENTED DATABASES

ADRIAN SERGIU DARABANT, ALINA CAMPAN, AND ANDREEA NAVROSCHI-SZASZ

Abstract. Distributed Object Oriented Databases require class fragmentation, performed either horizontally or vertically. Complex class relationships like aggregation and/or association are often represented as two-way references or object-links between classes. In order to obtain a good quality horizontal fragmentation, an optimal class processing order is needed. We present in this paper a new technique for establishing an order for class fragmentation. We improve fragmentation quality by capturing the semantic of input queries in the context of the aggregation hierarchy.

1. Introduction

Fragmentation is an important task that should be carried out in a Distributed Object Oriented Database (DOODB). The purpose of distributing a database is to increase query processing parallelism and to achieve high performance. Similar to the relational model, fragmentation in DOODB is performed horizontally and vertically. Horizontal fragmentation groups into fragments objects that are highly used together. Each object has the same structure and a different state or content. Thus, a horizontal fragment of a class contains a subset of the whole class extension. Vertical fragmentation breaks the logical structure of the class: *attributes* and *methods*, and distributes them across the fragments. The objective here is to group class attributes and methods that are frequently accessed together by queries. Each fragment contains, in this case, the same objects, but with different subsets of the attributes and methods [3].

Compared to the flat relational model, the object oriented paradigm introduces new issues into the fragmentation problem, due to its inherent complex nature. Complex object relationships like aggregation and association are part of this paradigm. They are often represented as two-way pointers or references, for performance reasons. This symmetric representation induces many cycles in

Received by the editors: 3/05/2004.

2000 *Mathematics Subject Classification.* C2.4, H2.4.

1998 *CR Categories and Descriptors.* H.2.4 [**DATABASE MANAGEMENT**]: Systems – *Distributed databases, Object-oriented databases* ; C.2.4 [**COMPUTER-COMMUNICATION NETWORKS**]: Distributed Systems – *Distributed databases* .

the association and aggregation graphs. We claim that the order of class fragmentation should take in account the links between classes. As long as there are cycles in the class graphs it is difficult to establish a fragmentation order between classes. In this paper we propose a method that properly eliminates cycles, by taking into consideration the semantic of class relationships and/or the strength of these relationships. We use query statistics in order to quantify the *strength* of links between entities. We propose an algorithmic approach that determines the proper order of fragmentation in an object database. Similar work has been conducted in [6], but only the inheritance relations are taken in account. The full semantic power of aggregation and associations is not captured however. Other research directions in object-oriented fragmentation do not address this problem at all [2, 5, 7, 8].

In section 2 we present the data model and basic concepts needed for problem definition. In section 3 we give an algorithm for establishing the optimal class fragmentation order in a context of an input set of queries. In section 4 we apply our algorithm to an example database and we conclude in section 5.

2. Data Model and Basic Concepts

We use an object-oriented model with the basic features described in the literature [1, 3]. Object-oriented databases represent data entities as objects supporting features like inheritance, encapsulation, polymorphism, etc. Objects with common attributes and methods are grouped into classes. A class is an ordered tuple $C = (K, A, M, I)$, where A is the set of object attributes, M is the set of methods, K is the class identifier and I is the set of instances of class C. Class attributes/methods are classified as simple and complex. Simple attributes have primitive data types as their domain. Simple methods access only attributes of their class. Complex attributes have other classes as their domain. Complex methods access attributes and/or methods of other classes. Further, they can have as return value objects of a different class type.

Every object in the database is uniquely identified by an OID. Each class can be seen in turn as a class object. Class objects are grouped together in metaclasses [3].

Classes are organized in an *inheritance hierarchy*, in which a subclass is a specialization of its superclass. Although we deal here for simplicity only with simple inheritance i.e. a class can have at most one superclass, moving to multiple inheritance would not affect the fragmentation algorithms in any way, as long as the inheritance conflicts are dealt with into the data model. We denote the fact that C_1 is a superclass of C_2 by $C_1 \prec C_2$. Association between an object and a class is materialized by the instantiation operation. An object O *is an instance of a class* C if C is the most specialized class associated with O in the inheritance hierarchy. An object O *is member of a class* C if O is instance of C or of one of subclasses of C.

An OODB is a set of classes from an inheritance hierarchy, with all their instances. There is a special class Root that is the ancestor of all classes in the database.

Aggregation and *association* are implemented as OID pointers. They are represented as a directed cyclic graph.

Definition 1. An ***entry point*** into a database is a meta-class instance bound to a known variable in the system.

An entry point allows navigation to all classes and class instances of its sub-tree (including itself). There are usually more entry points in an object database.

Definition 2. Given a complex hierarchy H, a ***path expression*** P is defined as $C_1.A_1...A_n$, $n \geq 1$ where: C_1 is an entry point in H, A_1 is an attribute of class C_1, A_i is an attribute of class C_i in H such that C_i is the domain of attribute A_{i-1} of class C_{i-1} $(1 \geq i \geq n)$.

Definition 3. A ***query*** is a tuple with the following structure, q=(Target class, Range source, Qualification clause), where:

- *Target class* - (query operand) specifies the root of the class hierarchy over which the query returns its object instances.
- *Range source* - a path expression specifying the source hierarchy.
- *Qualification clause* - logical expression over the class attributes in conjunctive normal form. The logical expression is constructed using simple predicates: *attribute* θ *value* where $q \in \{<, >, \leq, \geq, =, \neq, \}$.

Definition 4. We model the ***inheritance hierarchy*** as a directed acyclic graph Inh = (Class, Γ), where Class is the set of all classes in the database, $\Gamma = \{(C_1, C_2) | C_1, C_2 \in \text{Class}\ , C_1 \neq C_2, C_2 \text{ is superclass of } C_1\}$.

We give a working example of an inheritance hierarchy for a reduced university database in Fig. 1.

Let $Q = q_1, \ldots, q_t$ be the set of all queries in respect to which we want to perform the fragmentation.

Definition 5. We model the ***aggregation hierarchy*** as directed cyclic graph Agg = (Class, Λ), where Class is the set of all classes in the database, $\Lambda = \{(C_1, C_2) | C_1, C_2 \in \text{Class}, C_1 \neq C_2, C_2$ aggregates or is associated to $C_1\}$. Each edge u receives a *weight*, denoted as *weigth(u)*, equal to the number of path expression from queries in Q traversing that edge. We say that a link is *stronger* as its weight is larger.

The aggregation hierarchy for our example is depicted in Figure 2.

3. Class Fragmentation Ordering

In the following paragraphs we give a representation of all relations between classes (inheritance, aggregation and association), we explain the reasons behind

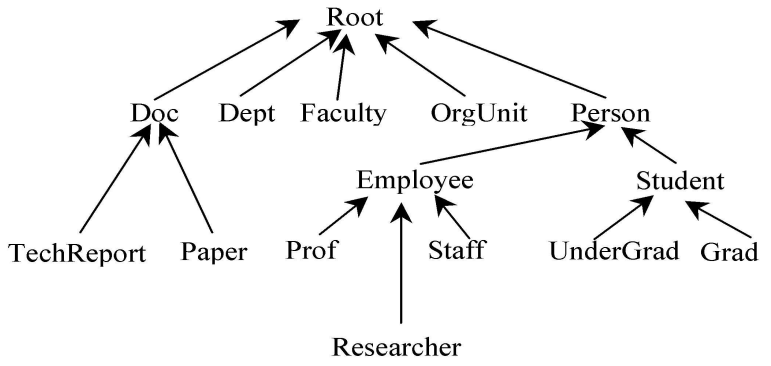

FIGURE 1. The database inheritance hierarchy

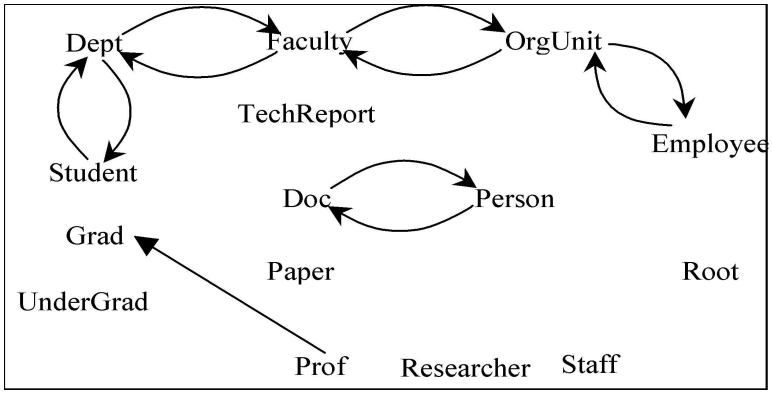

FIGURE 2. The database aggregation hierarchy

our modeling scheme. We propose an algorithm that takes as input an object database and returns the optimal order in which classes should be fragmented so that semantics of the queries is fully reflected in the resulting fragments. We prove that the proper order can always be found - i.e. the problem has always at least one solution.

3.1. **Relationship modeling.**

Definition 6. Let $Pred = \{p_1, \ldots, p_q\}$ be the set of all simple predicates Q is defined on. Let $Pred(C) = \{p \in Pred | p$ imposes a condition to an attribute of class C or to an attribute of class C', where $C' \prec C\}$.

Given two classes C' and C, where $C' \prec C$, $Pred(C) \supseteq Pred(C')$. Thus the set of predicates for class C comprises all the predicates directly imposed on attributes of C and the predicates defined on attributes of its parent class C' and inherited from it. We model class predicates this way in order to capture on subclasses the semantic of queries defined on superclasses [4].

Definition 7. The directed graph modeling all relations between classes is denoted as *CandidateRelGraph*=(NAClass,U) where NAClass is the set of non-abstract classes in Class. $(C_1, C_2) \in U$ if:

- Class C_1 is aggregated by class C_2
- $\exists C \in$Class, $C \prec C_2$ and C aggregates C_1; this means that aggregation is inherited by a class from its ancestor.
- $\exists C \in$Class, $C \prec C_1$ and C is aggregated by C_2; this means that the fact of being aggregated is inherited by a class from its ancestor.
- $\exists C'_1, C'_2 \in$Class, $C'_1 \prec C_1, C'_2 \prec C_2$ and C'_1 is aggregated by C'_2; i.e. the fact of being aggregated by / aggregating a class is inherited.

Definition 8. Let *RelGraph*=(FClass,V) be the subgraph of *CandidateRelGraph*, where FClass=NAClass-$\{C | C \in$ NAClass, $Pred(C) = \emptyset$ and $\nexists$ a path in *CandidateRelGraph* between C and a class $C' \in$NAClass so that $Pred(C') \neq \emptyset\}$.

We don't keep abstract classes because clustering an empty set of objects has no meaning. By propagating aggregation relations through the inheritance hierarchy we eliminate the inheritance dimension from *RelGraph* and we keep in the same time the semantic of aggregation for each particular class. This will turn out to be a necessary and helpful decision for our method, as we will see. As a consequence, every pair (inheritance path, aggregation link) is transformed into a link with the weight of the aggregation link.

We start from the presumption that a fragmentation algorithm first performs fragmentation on the classes with query conditions on them. Then, derived fragmentation is performed on the aggregator or aggregated classes that are linked with the already fragmented classes. Definition 8 eliminates classes that do not meet any of these requirements because they cannot be fragmented.

Definition 9. We denote by $CAN(C)$ the node aggregation coefficient of class C, defined as follows:

$$CAN(C) = \sum_{u \in V, u=(C,\_)} weigth(u) + CAN(C'), C' \prec C. \tag{1}$$

An aggregation edge $C_1 \rightarrow C_2$ will be traversed by path expressions following the pattern $a.b.c \ldots C2.C1 \ldots .d.e.f$. This means that the path expression navigates from instances of C_2 to one or more instances of C_1. It makes sense to first fragment class C_1 and then class C_2; when fragmenting C_2 we should take in

account the fragmentation of C_1. We want to place in the same fragment of C_2 objects aggregating instances from a fragment of C_1. Objects of a fragment of C_2 should aggregate as much as possible objects from the same fragment of C_1.

When the aggregation graph has cycles they should be broken in order to be able to perform fragmentation. One edge needs to be ignored from each cycle. Our decision is to select the least traversed edge by path expressions. The CAN measure precisely quantifies the navigation frequency for edges adjacent to a node.

When class A aggregates a class C_1 it aggregates all subclasses of C_1. A refers instances of C_1 and of all its subclasses. This is why we transfer the CAN of C_1 to its subclasses.

The CAN coefficient of a node quantifies how strong the other classes aggregate the node. Intuitively, classes with large CAN values strongly influence the overall fragmentation of the entire database. We want to fragment classes in descending order of CANs, as much as possible. We conserve this way the strongest aggregation relationships and we perform derived fragmentation with respect to these relationships, together with primary fragmentation, in a single step.

3.2. **Algorithm FragOrder.**

```
Algorithm FragOrder is
Input:  RelGraph=(FClass,V); CAN(C),where C ∈FClass;
Output:  L=[C1,C2,...,Cn], n=|FClass|; L gives the fragmentation order.
Var:
    LNC - CAN ordered (descending) list of classes having conditions;
    LNFC - the set of classes having no conditions imposed on them.
 Begin
    LNC:=[C|C∈FClass, Pred(C) ≠ ∅, in CAN descending order];
    LNFC:=FClass-LNC; L:=∅;
    While FClass≠ ∅ do
       i:=1; continue:=true;
       While i≤|LNC| and continue=true do
          C:=LNC[i];
          If {u|u∈V,u=(A,C),A∈LNC}=∅ //C doesn't aggregate any class
             Call Remove(C,RelGraph,LNC,LNFC);
             continue:=false;
          Else //C aggregates classes from LNC
             Call BreakCycles(C,RelGraph);
             If {u|u∈V,u=(CA,C),CA∈LNC}=∅
                Call Remove(C,RelGraph,LNC,LNFC)
                continue:=false;
             End if;
          End if;
```

```
      End while;
    End while;
End.

Subalgorithm Remove(C,RelGraph,LNC,LNFC) is
 Begin
    LNC:=LNC-[C]; L:=L+[C];
    New:={U|U∈LNFC,(C,U)∈V} ∪
         {U|U∈LNFC,(U,C)∈V, ∄ path in RelGraph from CC∈LNC to U};
    LNC:=LNC+[C|C∈New]; Resort(LNC);
    LNFC:=LNFC-New;
    FClass:=FClass-C;
End;

Subalgorithm BreakCycles(C,RelGraph) is
Begin
   // We first break trivial cycles
   @ While C participates in a trivial cycle P
      V:=V-{u|u∈P,weigth(u)=min{weight(v),v∈P}}
   @ End While
   // Breaking non-trivial cycles
   @ While C participates in a cycle P
      V:=V-{u|u∈P,weigth(u)=min{weight(v),v∈P}}
   @ End While
End;
```

We use LNC - the list of classes with conditions either directly defined on them or induced by fragments of their aggregated classes. These are the classes that we can fragment at a given moment. We try to take out from LNC the most aggregated class (this class should be the first to fragment) - and add this class to L. A class C (with maximum CAN) can be taken out from LNC if it has no incident edges (C does not aggregate any of the remaining classes in LNC). If C aggregates classes from LNC then we break the trivial cycles C is involved in (if any), in order to free it. In each cycle we break the weakest aggregation edge. If C is not freed then we pass to the next class in LNC and we reiterate the same process. When we remove a class from LNC, we add that class to L and we add all its successors and all its predecessors (S) that do not aggregate classes from LNC (there cannot be any induced aggregation conditions from LNC on S).

3.3. **Correctness Issues.**

We prove that the algorithm terminates and that the problem always has (at least one) solution. First of all we prove that when no more nodes can be taken

out from *RelGraph*, there is a cycle in *RelGraph*. Then we prove that, by breaking cycles, we free in each iteration one node.

Theorem 1. *Given an oriented graph* $G = (X, \Gamma)$, $X = \{x_1, \ldots, x_n\}$, *if* $\forall x_i$, $|\Gamma^-(x_i)| > 0 \Rightarrow G$ *contains at least a cycle.*

Proof: Let's presume by *reductio ad absurdum* that $\forall x_i, |\Gamma^-(x_i)| > 0$ and there are no cycles in G.
We take an $x_{i_1} \in X$. $|\Gamma^-(x_{i_1})| > 0$ means that there exists at least one $x_{i_2} \in X$ so that $(x_{i_2}, x_{i_1}) \in \Gamma$. Let Y be the set $\{x_{i_1}\}$. $|\Gamma^-(x_{i_2}) > 0$ means that there exists at least one $x_{i_3} \in X$ so that $(x_{i_3}, x_{i_2}) \in \Gamma$. If $x_{i_3} \in Y$ then there is a cycle. If $x_{i_3} \notin Y$, let $Y = Y \cup \{x_{i_2}\}$. By continuing this process, either we obtain a cycle if $x_{i_k} \in Y$, or we reach the situation where we detected no cycle and $Y = \{x_{i_1}, \ldots, x_{i_n}\}$. But $|\Gamma^-(x_{i_n})| > 0$, which means that there exists at least one $y \in X$ so that $(y, x_{i_n}) \in \Gamma$. Node y must be in Y, as there are no other nodes from which to choose. We have thus constructed a cycle, fact that contradicts our presumption.

Theorem 2. *Given an oriented graph* $G = (X, \Gamma)$, $X = \{x_1, \ldots, x_n\}$, *if* $\forall x_i$, $|\Gamma^-(x_i)| > 0$, *by breaking a finite number* N *of times an edge* (x_{k_1}, x_{k_2}), $1 \leq k \leq N$, *we obtain a new graph* $G' = (X, \Gamma'), \Gamma' = \Gamma - \{(x_{k_1}, x_{k_2}), 1 \leq k \leq N\}$ *so that exists* $x_j \in X$ *with* $|\Gamma'^-(x_j)| = 0$.

Proof: If $\forall x_i \in X, |\Gamma^-(x_i)| > 0$, then, according to Theorem 1, there is at least a cycle in G. Let $(x_{i_1}, x_{i_2}, \ldots, x_{i_m}), m \leq n, x_{i_j} \neq x_{i_{j+1}}, 1 \leq j < m, x_{i_m} = x_{i_1}$ be a cycle. We choose from it an edge to be eliminated, let it be $(x_{i_j}, x_{i_{j+1}}), 1 \leq j < m$. This means that $|\Gamma^-(x_{i_{j+1}})|$ decreases by 1. As $|\Gamma^-(x_i)|$ is finite $\forall x_i \in X$, following the same procedure a finite number of times we reach the situation when for a node $x_j \in X, |\Gamma^-(x_j)| = 0$.

4. An example

Given the database in Figure 1 and Figure 2 having as entry points Doc, Person, Faculty, and a set of queries:
q1: This application retrieves all graduates having their supervisor in ProgrMeth or InfSyst OrgUnits.
q1 = (Grad, Faculty.Dept.Student, Grad.Supervisor.OrgUnit.Name in ("ProgrMeth", "InfSyst"));
q2: This application retrieves all undergraduates with scholarships from Comp. Sci.
q2 = (UnderGrad, Faculty.Dept.Student, UnderGrad.Dept.Name like "CS%" and UnderGrad.Grade between 7 and 10)
q3: This application retrieves all undergraduates older than 24 years from Math and Comp. Sci. depts.
q3 = (UnderGrad, Faculty.Dept.Student, (UnderGrad.Dept.Name like "Math%" or UnderGrad.Dept.Name like "CS%") and UnderGrad.Age()≥24)
q4: This application retrieves all researchers having published at least two papers.

q4 = (Researcher, Doc.Person,Researcher.count(Reasercher.doc)≥2)
q5: This application retrieves all teachers from ProgrMeth and InfSyst OrgUnits with salary greater than 40000.
q5 = (Prof, Faculty.OrgUnit.Employee, Prof.OrgUnit.Name in ("ProgrMeth", "InfSyst") and Prof.salary≥40000)
q6: This application retrieves all profs having published at IEEE or ACM
q6 = (Prof, Doc.Person., Prof.Paper.Publisher in ("IEEE" , "ACM") and Prof.Position="prof")
q7: This application retrieves all tech reports published after 1999.
q7 = (TechReport, Doc, TechReport.year>1999)
q8: This application retrieves all depts with students having grades less than 5.
q8 = (Set(Student.Dept), Person, Student.Grade<5)
q9: This application retrieves all employees with salaries greater than 35000.
q9 = (Employee, Person, Employee.salary>35000)
q10: This application retrieves all grads with at least one paper.
q10 = (Grad, Person, Grad.count(Grad.Paper)≥1)
q11: This application retrieves all students from Comp. Sci. depts
q11 = (Student, Person,Student.Dept.Name like "CS%")
q12: This application retrieves all students from Math depts.
q12 = (Student, Person,Student.Dept.Name like "Math%")
q13: This application retrieves all staff with salaries greater than 12000.
q13 = (Staff, Person, Staff.salary>12000)

Aggregation Edges LineLabel → ColLabel	Dept – CAN=10	Grad – CAN=7	UnderGrad - CAN=7	Prof – CAN=6	Researcher– CAN=5	Staff– CAN=5	OrgUnit– CAN=3	TechReport– CAN=2	Paper– CAN=2	Faculty– CAN=0
Dept		2	5							3
Grad	3							2	2	
UnderGrad	3							2	2	
Prof		1					1	2	2	
Researcher							1	2	2	
Staff							1	2	2	
OrgUnit				2	0	0				1
TechReport		1		1	0					
Paper		1		1	0					
Faculty	0						0			

TABLE 1. RelGraph adjacency matrix.

the adjacency matrix for *RelGraph* is given in Table 1.

In our example all non-abstract classes have conditions imposed on them by the set of queries. The resulting class fragmentation order resulted by applying the algorithm is: Researcher→Staff→OrgUnit→Prof→Grad→Dept→UnderGrad →TechReport→Paper→Faculty.

5. Conclusions and Future Work

We claim that class fragmentation order is significant in distributed object orientated databases. We investigate in this paper a method for choosing this order in respect to a set of application queries given as input. Initial experiments show that the fragmentation order has an important impact in the fragmentation quality. We plan to develop measures for quantifying this quality improvement. We also aim to compare different, richer scenarios with and without fragmentation order involved and to study the limitations, if any, of this technique.

References

[1] Atkinson, M., Bancilhon, F., DeWitt, D., Dittrich, K., Maier, D., Zdonik, S. - The Object Oriented Database Manifesto, In Proc. of the 1st Int. Conf. on Deductive and Object-Oriented Databases, 1989.

[2] Baiao, F., Mattoso, M. - A Mixed Fragmentation Algorithm for Distributed Object Oriented Databases, In Proc. Of the 9th Int. Conf. on Computing Information, Canada, pp 141-148, 1998.

[3] Bertino, E., Martino, L. - Object-Oriented Database Systems; Concepts and Architectures, Addison-Wesley, 1993.

[4] Bellatreche,L., Karlapalem, K., Simonet, A. - Horizontal Class Partitioning in Object-Oriented Databases, In Lecture Notes in Computer Science, volume 1308, pp 58-67, Toulouse, France, 1997.

[5] Darabant, A.S, Campan, A. - Horizontal object fragmentation using clustering techniques, In Proc. of Computers and Communications, Oradea, Romania , 2004(to appear).

[6] Ezeife, C.I., Barker, K. - A Comprehensive Approach to Horizontal Class Fragmentation in a Distributed Object Based System, International Journal of Distributed and Parallel Databases, 3(3), pp 247-272, 1995.

[7] Ezeife, C.I., Barker, K. - Horizontal Class Fragmentation for Advanced-Object Modes in a Distributed Object-Based System, In the Proceedings of the 9th International Symposium on Computer and Information Sciences, Antalya, Turkey, pp 25-32, 1994.

[8] Ravat, S. - La fragmentation d'un schema conceptuel oriente objet, In Ingenierie des systemes d'information (ISI), 4(2), pp 161-193, 1996.

Babes Bolyai University, Cluj Napoca,Romania
E-mail address: dadi@cs.ubbcluj.ro

Babes Bolyai University, Cluj Napoca, Romania
E-mail address: alina@cs.ubbcluj.ro

Babes Bolyai University, Cluj Napoca, Romania
E-mail address: deiush@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

A WEIGTHED-PATH-FOLLOWING METHOD FOR THE LINEAR COMPLEMENTARITY PROBLEM

MOHAMED ACHACHE

Abstract. In a recent paper [3] a weighted path-following interior point method (IPM) has been developed to solve linear programs (LP) based on a method for finding a new family of search directions. In this paper we describe a similar approach for linear complementarity problems (LCP). We prove that the algorithm performs the same number of iterations as in [3] .

1. Introduction

Formally, generalized path-following interior point methods (or the so-called target-following methods) are related to the classical central path methods but they are more general in the sense that the barrier parameter is a multidimensional vector and not a real number. Geometrically, these methods are based on the observation that with every algorithm which follows the central path we can associate a target sequence on this central path. A good survey of this concept can be found in [9] . Weighted-path following methods can be seen as a particular case of target-following methods. These methods were studied by Ding and Li [4] for primal-dual linear complementarity problems. Recently, Darvay [3] has been developed a weighted path-following algorithm for solving the linear optimization (LO) problem, based on a new method for finding a new family of search directions. In this paper, we describe a similar approach for solving the linear complementarity problem (LCP).

Received by the editors: December 10, 2003.

1991 *Mathematics Subject Classification.* 90C33, 90C51.

Key words and phrases. Linear complementarity problem, Weighted-path following method, Primal-dual algorithm.

I express my appreciation for the helpful suggestions made by the referees. I also thank Zsolt Darvay for helping me with several suggestions.

This paper is organized as follows. In the next section the LCP problem, the statement of the problem and the associated weighted problem are presented. Section 3 deals with the existence and uniqueness of the solution of the weighted problem and its differentiability. Section 4 is devoted to the search direction and the description of the algorithm. Section 5 presents the convergence analysis of the algorithm and its polynomial complexity. Section 6 ends the paper with a conclusion.

1.1. **Notations.** Our notation is the usual one. In particular, $\Re^n$ denotes the space of real $n-$dimensional vectors and $\Re^n_+$ the nonnegative orthant of $\Re^n$. Let $u,\ v \in \Re^n,\ u^T v$ is their inner product, $\| u\|$ is the Euclidean norm and $\| u\|_\infty$ is maximum norm. Given a vector u in $\Re^n$, U =diag(u) is the $n \times n$ diagonal matrix with $U_{ii} = u_i$ for all i, where U_{ii} denotes the i-th element on the diagonal of U. The vector $e = [1, \ldots, 1]^T$ is the vector of ones in $\Re^n$. Given the vectors x and y in $\Re^n$, $xy = [x_1y_1, x_2y_2, \ldots, x_ny_n]^T$ denotes the coordinate-wise product of x and y, and $\langle x, y\rangle$ the scalar product of x and y. We shall use also the notation $\frac{x}{y} = \left[\frac{x_1}{y_1}, \frac{x_2}{y_2}, \ldots, \frac{x_n}{y_n}\right]^T$ with $y_i \neq 0$ for all i. For a given arbitrary function ψ, and an arbitrary vector x we will use the notation $\psi(x) = [\psi(x_1), \ldots, \psi(x_n)]^T$. $d(A, B)$ is the distance between the sets A and B.

2. The LCP problem and the statement of the problem

The linear complementarity problem (LCP) is defined as:
find $x \geqslant 0$ and $y \geqslant 0$ such that

$$y = Mx + q,\ x^T y = 0 \tag{1}$$

where M is a given $(n \times n)$ real matrix and q is a given $n-$dimensional real vector, the inequalities are understood to be components-wise. The complementarity condition $x^T y = 0$, is equivalent to $x_i y_i = 0$, for $i = 1, 2, ..., n$.
The linear complementarity problems model many important mathematical problems. The books [1, 7] are good documentations of complementarity problems.
The feasible set, the strict feasible set and the solution set of the

problem (1) are denoted respectively by

$$\begin{aligned} \mathcal{F} &= \left\{(x,y) \in \mathcal{R}^2 : Mx + q = y,\ x \geqslant 0,\ y \geqslant 0\right\}, \\ \mathcal{F}^0 &= \left\{(x,y) \in \mathcal{F} :\ x > 0,\ y > 0\right\}, \end{aligned}$$

and

$$\mathcal{S}_{cp} = \left\{(x,y) \in \mathcal{F} : x_i y_i = 0, i = 1, 2, ..., n\right\}.$$

Throughout the paper we make the following assumptions.
Assumption 1 $\mathcal{F}^0 \neq \emptyset$.
Assumption 2 M is a positive semidefinite matrix.
Assumption 1 implies that $\mathcal{F}^0$ is the relative interior of $\mathcal{F}$ and also that the set of solution of (LCP) is nonempty convex and compact.
We formulate (1) into the equivalent convex minimization problem:

$$\min \left[x^T y \ \text{ s.t. } x \geqslant 0,\ y \geqslant 0, Mx + q = y\right]. \tag{2}$$

We observe, that if (x, y) is a solution of the (LCP), then the global minimizer of (2) is zero, see [13]. We associate with (2) the following weighted problem:

$$(\mathrm{P}_w) \qquad \min \left[g_w(x,y) = x^T y - \sum_{i=1}^{n} w_i^2 \ln \frac{x_i y_i}{w_i^2}\right], \quad \text{s.t. } (x,y) \in \mathcal{F}^0.$$

where $w^2 = [w_1^2, w_2^2, \ldots, w_n^2]^T$ for a given positive vector w.
Denote by $\mathcal{L}(x, y, z, w)$, the Lagrangian of the problem (P_w)

$$\mathcal{L}(x,y,z,w) = x^T y + \sum_{i=1}^{n} w_i^2 \ln\left(\frac{w_i^2}{x_i y_i}\right) - z^T(Mx + q - y). \tag{3}$$

The first order optimality conditions for (3) give the system of nonlinear equations

$$y - X^{-1}w^2 - M^T z = 0, \tag{4}$$

$$x - Y^{-1}w^2 + z = 0, \tag{5}$$

$$Mx + q - y = 0, \tag{6}$$

where $z \in \mathfrak{R}^n$, $X = \text{diag}\,(x)$ and $Y = \text{diag}(y)$.
From the first and the second equation in $(4-6)$, we obtain

$$\begin{cases} XYe - w^2 - XM^T z = 0, \\ XYe - w^2 + Yz = 0. \end{cases}$$

It follows that

$$(M^T + X^{-1}Y)z \;=\; 0. \tag{7}$$

Recall that M is positive semidefinite. Hence, for all $h \neq 0$

$$\langle \left(M^T + X^{-1}Y\right) h, h\rangle = \langle Mh, h\rangle + \langle X^{-1}Yh, h\rangle > \langle Mh, h\rangle \geqslant 0.$$

It follows that $\left(M^T + X^{-1}Y\right)$ is invertible. Thus $z = 0$, and the system (4-6) reduces to

$$\mathfrak{F}(x, y, w) = \begin{pmatrix} XYe - w^2 \\ Mx + q - y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{8}$$

The system (8) can be written as

$$Mx + q = y, \quad xy = w^2, \tag{9}$$

where $xy = [x_1y_1, x_2y_{2,\cdots}, x_ny_n]^T$. This form is convenient for the analysis of the convergence of the suggested algorithm. Hence, solving the problem (P_w) is equivalent to solving the system (9).

Our next aim is to show that the weighted problem (P_w) has one unique minimizer. To this end, we follow the technique of proof from [2, 10] for the classical barrier logarithmic methods to show the existence of the minimizer.

3. Existence and the uniqueness of the minimizer of (P_w)

Let us define the following function $\tilde{g}_w$ by

$$\tilde{g}_w(x) = g_w(x, Mx + q).$$

Proposition 3.1. *Under assumptions **1** and **2**, we have*
1) the function $\tilde{g}_w$ is strictly convex on

$$\{(x, y) : x > 0,\ y = Mx + q, y > 0\}.$$

2) the problem (P_w) and the system (8) are equivalent for $w > 0$.
3) for each $w \in (0, +\infty)^n$, the problem (P_w) has one unique global minimizer or equivalently the system (8) or (9) has a unique solution denoted by $(x(w), y(w))$ with $x(w) > 0$ and $y(w) > 0$.

Proof. To prove the strict convexity of $\tilde{g}_w$, we omit first the constant part $\sum_{i=1}^{n} w_i^2 \ln w_i^2$ from it. Then we have

$$\tilde{g}_w(x) = x^T(Mx+q) - \sum_{i=1}^{n} w_i^2 \ln x_i - \sum_{i=1}^{n} w_i^2 \ln(Mx+q)_i,$$

since

$$\nabla \tilde{g}_w(x) = (M+M^T)x + q - X^{-1}w^2 - M^T \left[\operatorname{diag}(Mx+q)\right]^{-1} w^2,$$

and

$$\nabla^2 \tilde{g}_w(x) = M + M^T + X^{-2}W + M^T \left[\operatorname{diag}(Mx+q)\right]^{-2} W^2 M$$

where $W^2 = \operatorname{diag}(w^2)$.
Hence $\nabla^2 \tilde{g}_w$ is a positive definite matrix and thereby $\tilde{g}_w$ is strictly convex function on $\{(x,y) : x > 0, y = Mx+q,\ y > 0\}$.
For the second statement, we have the objective function g_w of the problem (P_w) can be written as

$$\sum_{i=1}^{n} \left[x_i y_i - w_i^2 \ln \frac{x_i y_i}{w_i^2} \right]$$

and since each term in brackets attains the minimum under the condition $(x,y) \in \mathcal{F}^0$, if and only if, $x_i y_i = w_i^2$, then any minimizer of (P_w) is a solution of the system (8) and vice versa.
For the last statement, let $w \in (0,+\infty)^n$ be fixed. We have

$$\hat{g}_w(x) = g_w(x, Mx+q) = x^T(Mx+q) + \sum_{i=1}^{n} w_i^2 \ln \frac{x_i(Mx+q)_i}{w_i^2}.$$

Recall that in view of assumption 1, $\mathcal{S}_{cp}$ is nonempty and bounded. Therefore $\left\{(x,y) \in \mathcal{F} : x^T(Mx+q) \leqslant 0\right\}$ is bounded. By assumption 2, the level set $\left\{(x,y) \in \mathcal{F} : x^T(Mx+q) \leqslant t\right\}$ is bounded for any t. It follows that the level set

$$\Omega(t) = \{x \in \mathfrak{R}^n :\ g_w(x, Mx+q) \leqslant t,\ Mx+q > 0\}$$

is also bounded since $\hat{g}_w$ differs only by the term

$$-\sum_{i=1}^{n} w_i^2 \ln \frac{x_i(Mx+q)_i}{w_i^2}$$

from the quadratic function $x^T(Mx+q)$. Again in view of assumption 1, it follows that $\Omega(t) \neq \emptyset$ for sufficiently large t. Finally, if $(x,y) \in \mathcal{F}^0$ approaches the boundary of $\mathcal{F}$, then

$$\ln \frac{x_i(Mx+q)_i}{w_i^2} \to +\infty.$$

This implies that $g_w(x, Mx+q) \to +\infty$. Thus $\hat{g}_w$ must have a global minimizer in $\mathcal{F}^0$denoted by $x(w)$. Since $\hat{g}_w$ is strictly convex, then $x(w)$ is unique.
If one of the components of $x(w)$ is zero, then $g_w(x, Mx+q) \to +\infty$. It follows that $x(w) > 0$. From the system (8), $y(w)$ is also determined uniquely and $y(w) > 0$. ■
The next objective is to show the differentiability of the solutions of the weighted problem (P_w), $w \mapsto x(w)$ and $w \mapsto y(w)$ on $(0,\infty)^n$.

Theorem 3.2. *The functions $w \mapsto x(w)$ and $w \mapsto y(w)$ are $\mathcal{C}^\infty$ on $(0,\infty)^n$.*

Proof. Let us recall again the mapping defined in (8) by

$$\begin{aligned} \mathfrak{F} &: \mathfrak{R}_+^{2n} \times \mathfrak{R}_+^n \to \mathfrak{R}^{2n} \\ \mathfrak{F}(u,w) &= (Mx+q-y, Xy-w^2). \end{aligned}$$

The Fréchet derivative of $\mathfrak{F}$ with respect to $u=(x,y)$ is:

$$\mathfrak{F}'_u(u,w) = \begin{pmatrix} Y & X \\ M & -I \end{pmatrix}.$$

Since

$$\begin{pmatrix} I & 0 \\ -MY^{-1} & I \end{pmatrix} \begin{pmatrix} Y & X \\ M & -I \end{pmatrix} = \begin{pmatrix} Y & X \\ 0 & -(I+MY^{-1}X) \end{pmatrix},$$

it follows that the Jacobian matrix $\mathfrak{F}'_u(u,w)$ is invertible for (u,w) with $x>0, y>0$ and $w>0$.
Let now $\bar{w}$ be fixed in $(0,+\infty)^n$ and $u(\bar{w}) = 0$. Hence $\mathfrak{F}(\bar{u},\bar{w}) = 0$. Since $\mathfrak{F}$ is continuously differentiable and $\mathfrak{F}'(u,w)$ is invertible, applying the implicit function theorem we obtain that there exists θ continuously differentiable function on a neighborhood of $\bar{w}$ such that $\mathfrak{F}(\theta(w),w) = 0$. Since the nonlinear system (8) characterizes the optimal solution $(x(w),y(w))$ of (P_w), therefore $\theta(w) = u(w)$. The function u is differentiable. By an immediate induction it is C^∞. ■

Proposition 3.3. *Let $(x(w),\, y(w))$ be a solution of (P_w). Then*
1) $x^T(w)y(w) \to 0$ as $w \mapsto 0$.
2) $d(\mathcal{S}_\varepsilon, \mathcal{S}_{cp}) \to 0$ as $\varepsilon \mapsto 0$ where

$$\mathcal{S}_\varepsilon = \left\{(x,y) : x \geqslant 0,\ y \geqslant 0,\ Mx+q=y,\ x^Ty \leqslant \varepsilon\right\}.$$

3) combining 1) and 2) we have $d((x(w),y(w)), \mathcal{S}_{cp}) \to 0$ as $w \to 0$.

Theorem 3.4. *For any $w > 0$, there is a unique solution $(x(w), y(w))$ to (8) and the path $\{(x(w), y(w)) : w > 0\}$ is smooth.*

Remark 3.1. *The central path method corresponds to the path $\{(x(\mu e),\ y(\mu e)) :\ \mu > 0\}$. So, our method can be viewed as a generalization of the classical central path method.*

In the next section we describe a similar approach as in [3] to solve the (LCP). This section follows closely the argument developed in [3] for finding a new family of search directions by using the system (9).

4. New search directions and the algorithm

Let $\mathfrak{R}^+ = \{x \in \mathfrak{R}\ : x \geqslant 0\}$, and consider the function

$$\varphi \in \mathcal{C}^1,\ \varphi\ :\ \mathfrak{R}^+ \to \mathfrak{R}^+.$$

We suppose that φ is a one to one function, i.e. φ^{-1} exists. Then the system (9) can be written in the following equivalent form

$$\begin{aligned} Mx+q=y, \quad x \geqslant 0,\ y \geqslant 0,\\ \varphi(xy) = \varphi(w^2). \end{aligned} \tag{10}$$

Suppose that we have $(x,y) \in \mathcal{F}^0$, i.e. x and y are strictly feasible. Applying Newton's method for the system (10) we obtain the new class of search directions

$$\begin{aligned} M\Delta x = \Delta y,\\ y\varphi^{'}(xy)\Delta x + x\varphi^{'}(xy)\Delta y = \varphi(w^2) - \varphi(xy). \end{aligned} \tag{11}$$

Now the following notations are useful for studying the complexity of the proposed algorithm:

$$v = \sqrt{xy} \quad \text{and} \quad d = \sqrt{xy^{-1}}.$$

Observe that these notations lead to

$$d^{-1}x = dy = v. \tag{12}$$

Denote

$$d_x = d^{-1}\Delta x, \quad d_y = d\Delta y,$$

and hence, we have

$$v(d_x + d_y) = y\Delta x + x\Delta y, \tag{13}$$

and

$$d_x d_y = \Delta x \Delta y. \tag{14}$$

So the system (11) becomes

$$\begin{aligned} \bar{M}d_x &= d_y, \\ d_x + d_y &= p_v, \end{aligned}$$

where $\bar{M} = DMD$, with $D =$diag(d) and

$$p_v = \frac{\varphi(w^2) - \varphi(xy)}{v\varphi'(v^2)}.$$

As in [3] , we put $\varphi(t) = \sqrt{t}$. Hence the Newton's direction in (11) is

$$\begin{aligned} M\Delta x &= \Delta y, \\ \sqrt{\frac{y}{x}}\Delta x + \sqrt{\frac{x}{y}}\Delta y &= 2(w - \sqrt{xy}), \end{aligned} \tag{15}$$

with

$$p_v = 2(w - \sqrt{xy}) = 2(w - v). \tag{16}$$

We define for any vector v the following proximity measure by

$$\sigma(v, w) = \frac{\|p_v\|}{2\min(w)} = \frac{\|v - w\|}{\min(w)}, \tag{17}$$

where $\|.\|$ is the Euclidean norm and $\min(w) = \min\{w_i : 1 \leqslant i \leqslant n\}$. Now, for measuring the closeness of w^2 to the central path, we use the following quantity

$$\sigma_c(w) = \frac{\max(w^2)}{\min(w^2)},$$

where $\max(w) = \max\{w_i : 1 \leqslant i \leqslant n\}$. Now the primal-dual algorithm can be defined formally as follows.

Algorithm 4.1. We assume that $(x^0, y^0) \in \mathcal{F}^0$, and let $w^0 = \sqrt{x^0 y^0}$. Let $\epsilon > 0$ be the given tolerance, and $0 < \theta < 1$ the update parameter (default $\theta = 1/(\sigma_c(w^0)n)^{1/5}$).

begin
$x := x^0; y := y^0$
$w := w^0;$
while $x^T y > \epsilon$ do **begin**
$w := (1-\theta)w;$
compute $(\Delta x, \Delta y)$ from (15)
$x := x + \Delta x;$
$y := y + \Delta y;$
end
end.

In the following section we prove that the algorithm converges to a solution of the (LCP) in polynomial time.

5. The convergence analysis and the complexity analysis

Let

$$q_v = d_x - d_y.$$

Then

$$d_x d_y = \frac{p_v^2 - q_v^2}{4},$$

and

$$\|q_v\| \leqslant \|p_v\|. \tag{18}$$

This last result follows directly from the equality

$$\|p_v\|^2 = \|q_v\|^2 + 4 d_x^T d_y.$$

In the following Lemma we give a condition to ensure the feasibility of the full step Newton.
Let $x_+ = x + \Delta x$ then $y_+ = M(x+\Delta x)+q = Mx+q+M\Delta x = y+\Delta y$.

Lemma 5.1. *Let $\sigma = \sigma(u,v) < 1$. Then the full Newton step is strictly feasible, hence*

$$x_+ > 0 \text{ and } y_+ > 0.$$

Proof. For each $0 \leqslant \alpha \leqslant 1$ let $x_+(\alpha) = x + \alpha\Delta x$ and $y_+(\alpha) = y + \alpha\Delta y$. Hence

$$x_+(\alpha) y_+(\alpha) = xy + \alpha(x\Delta y + y\Delta x) + \alpha^2 \Delta x \Delta y.$$

Now, in view of (13) and (14) we have

$$x_+(\alpha) y_+(\alpha) = v^2 + \alpha v(d_x + d_y) + \alpha^2 d_x d_y.$$

In addition from (16) we have

$$v + \frac{p_v}{2} = w,$$

and thus

$$v^2 + vp_v = w^2 - \frac{p_v^2}{4}.$$

Thereby

$$\begin{aligned} x_+(\alpha)y_+(\alpha) &= (1-\alpha)v^2 + \alpha(v^2 + vp_v) + \frac{\alpha^2}{4}(p_v^2 - q_v^2) = \\ &= (1-\alpha)v^2 + \alpha(w^2 - (1-\alpha)\frac{p_v^2}{4} - \alpha\frac{q_v^2}{4}), \end{aligned} \tag{19}$$

thus the inequality $x_+(\alpha)y_+(\alpha) > 0$ holds if

$$\left\| (1-\alpha)\frac{p_v^2}{4} + \alpha\frac{q_v^2}{4} \right\|_\infty < \min(w^2).$$

Using (17) and (18) we get

$$\left\| (1-\alpha)\frac{p_v^2}{4} + \alpha\frac{q_v^2}{4} \right\|_\infty \leqslant (1-\alpha)\left\| \frac{p_v^2}{4} \right\|_\infty + \alpha\left\| \frac{q_v^2}{4} \right\|_\infty \leqslant$$

$$\leqslant (1-\alpha)\frac{\|p_v\|^2}{4} + \alpha\frac{\|q_v\|^2}{4} \leqslant \frac{\|p_v\|^2}{4} = \sigma^2 \min(w^2) < \min(w^2).$$

Hence, $x_+(\alpha)y_+(\alpha) > 0$ for each $0 \leqslant \alpha \leqslant 1$. Since $x_+(\alpha)$ and $y_+(\alpha)$ are linear functions of α, then they don't change sign on the interval $[0, 1]$ and for $\alpha = 0$ we have $x_+(0) > 0$ and $y_+(0) > 0$. This leads to $x_+(1) > 0$ and $y_+(1) > 0$. ■

In the next Lemma we show that $\sigma < 1$ is sufficient for the quadratic convergence of the Newton process.

Lemma 5.2. *Let $x_+ = x + \Delta x$ and $y_+ = y + \Delta y$ be the iterates obtained after a full Newton step with $v = \sqrt{xy}$ and $v_+ = \sqrt{x_+ y_+}$. Suppose $\sigma = \sigma(v, w) < 1$. Then*

$$\sigma(v_+, w) \leqslant \frac{\sigma^2}{1 + \sqrt{1 - \sigma^2}}.$$

Thus $\sigma(v_+, w) < \sigma^2$,which means quadratic convergence of the Newton step.

Proof. By substituting $\alpha = 1$ in (19) we have

$$v_+^2 = w^2 - \frac{q_v^2}{4}. \tag{20}$$

Using (13) and (20) we obtain

$$\begin{aligned} \min(v_+^2) &\geqslant \min(w^2) - \frac{\|q_v^2\|_\infty}{4} \geqslant \min(w^2) - \frac{\|q_v\|^2}{4} \geqslant \\ &\geqslant \min(w^2) - \frac{\|p_v\|^2}{4} = \min(w^2)(1-\sigma^2), \end{aligned}$$

and this relation yields

$$\min(v_+) \geqslant \min(w)(\sqrt{1-\sigma^2}). \tag{21}$$

Furthermore, from (17) and (21) we get

$$\begin{aligned} \sigma(v_+, w) &= \frac{\|w - v_+\|}{\min(w)} = \frac{1}{\min(w)} \left\| \frac{w^2 - v^2}{w + v^+} \right\| \leqslant \\ &\leqslant \frac{\left\|w^2 - v_+^2\right\|}{\min(w)(\min(w) + \min(v_+)} \leqslant \frac{\|p_v^2\|}{(2\min(w))^2(1 - \sqrt{1-\sigma^2})} \\ &\leqslant \frac{1}{1-\sqrt{1-\sigma^2}} \left(\frac{\|p_v\|}{2\min(v)} \right)^2 = \frac{\sigma^2}{1-\sqrt{1-\sigma^2}}. \end{aligned}$$

This proves the lemma. ■ In the following lemma we find an upper bound for the duality gap obtained after a full Newton step.

Lemma 5.3. *Let $x_+ = x + \Delta x$ and $y_+ = y + \Delta y$. Then the duality gap is*

$$x_+^T y_+ = \left\|w^2\right\| - \frac{\|q_v\|^2}{4},$$

hence

$$x_+^T y_+ \leqslant \left\|w^2\right\|.$$

Proof. In view of (19) and with $\alpha = 1$, we have

$$x_+ y_+ = w^2 - \frac{q_v^2}{4},$$

then

$$x_+^T y_+ = e^T(x_+ y_+) = e^T w^2 - \frac{e^T q_v^2}{4} = \left\|w^2\right\| - \frac{\|q_v\|^2}{4}.$$

Hence

$$x_+^T y_+ \leqslant \left\| w^2 \right\|.$$

The proof is complete. ■

The next Lemma discusses the influence on the proximity measure of the Newton process followed by a step along the weighted path.

Lemma 5.4. [3] *Let $\sigma = \sigma(v,w) < 1$ and $w_+ = (1-\theta)w$, where $0 < \theta < 1$. Then*

$$\sigma(v_+, w_+) \leqslant \frac{\theta}{1-\theta}\sqrt{\sigma_c(w)n} + \frac{1}{1-\theta}\sigma(v_+, w).$$

Furthermore, if $\sigma \leqslant 1/2$, $\theta = 1/(\sigma_c(w)n)^{1/5}$ and $n \geqslant 4$ then we get $\sigma(v_+, w_+) \leqslant 1/2$.

Lemma 5.5. [3] *Assume that x^0 and y^0 are strictly feasible, and let $w^0 = \sqrt{x^0 y^0}$. Moreover, let x^k and y^k be the vectors obtained after k iterations. Then the inequality $\left(x^k\right)^T y^k \leqslant \epsilon$ is satisfied for*

$$k \geqslant \left\lceil \frac{1}{2\theta} \ln \frac{(x^0)^T y^0}{\epsilon} \right\rceil.$$

For the default $\theta = 1/(\sigma_c(w^0)n)^{1/5}$ we obtain the following theorem.

Theorem 5.6. [3] *Suppose that the pair $(x^0, y^0) \in \mathcal{F}^0$, and let $w^0 = \sqrt{x^0 y^0}$. If $\theta = 1/(\sigma_c(w^0)n)^{1/5}$ then Algorithm 4.1 requires at most*

$$\left\lceil \frac{5}{2}\sqrt{\sigma_c(w^0)n} \ln \frac{(x^0)^T y^0}{\epsilon} \right\rceil$$

iterations. For the resulting vectors we have $\left(x^k\right)^T y^k \leqslant \epsilon$.

6. Conclusion

In this paper, we have described a similar approach to the one developed by Darvay for linear programs, to solve the monotone linear complementarity problem. Here we have transformed the (LCP) problem into an equivalent convex minimization problem based on some weighted methods. We have adopted the function developed in [3] to obtain new search directions and to develop the new primal-dual algorithm. We have proved that this algorithm performs no more than

$$\left\lceil \frac{5}{2}\sqrt{\sigma_c(w^0)n} \ln \frac{(x^0)^T y^0}{\epsilon} \right\rceil$$

iterations. This result is the same as the one found for the algorithm developed for the (LP).

References

[1] R.W. Cottle, J.S. Pang, and R.E. Stone. (1992): The linear complementarity problem. Academic Press, San Deigo.

[2] Zs. Darvay.(2002): A new algorithm for solving self-dual linear programming problems. *Studia Universitatis Babes-Bolyai, Series Informatica,* **47**(2), 15-26.

[3] Zs. Darvay.(2002): A weighted-path-following method for linear optimization. *Studia Universitatis Babes-Bolyai, Series Informatica,* **47**(1), 3-12.

[4] J. Ding, and T.Y. Li. (1996): An algorithm based on weigthed barrier functions for linear complementarity problems. *Arabian Journal for Science and Engineering.* **15,** 679-685.

[5] J. Gondzio. (1998): Warm start of the primal-dual applied to the cutting plane scheme. *Mathematical Programming* **83**, 125-143.

[6] C. Gonzaga. (1992): Path following methods in linear programming. *SIAM Review.* **34**, 167-242.

[7] K.G. Murty. (1988): Linear complementarity, Linear and nonlinear programming. Vol **3** of Sigma Series in Applied Mathematics, Heldermann Verlag, Berlin, Germany.

[8] B. Jansen, C. Roos, and T. Terlaky.(1993): A family of polynomial affine scaling algorithms for positive semi-definite linear complementarity problems. Technical Report DUT-TWI-93-112, Delft, The Netherlands.

[9] B. Jansen, C. Roos, T. Terlaky, and J.Ph. Vial. (1996): Primal-dual target following algorithms for linear programming. *Annals of operations Research*, **62**, 197-231.

[10] M. Kojima, S. Mizuno, and A. Yoshize. (1989): A polynomial-time algorithm for a class of linear complementarity problems. *Mathematical Programming* **44**, 1-26.

[11] M. Kojima, N.Meggido, T. Noma, and A. Yoshize.(1991): A unified approach to Interior Point Algorithms for Linear Complementarity problems, volume 538 of Lecture Notes in Computer Science. Spring Verlag, Berlin, Germany.

[12] Y. Zhang. (1994): On the convergence of a class of infeasible interior point methods for the horizontal linear complementarity problems. *SIAM, Journal of Optimization*, **4**, 53-67.

[13] S.J. Wright. (1997): Primal-dual interior point methods, SIAM, Philadelphia.

Département de Mathématiques, Faculté des Sciences, Université Ferhat Abbas de Sétif, Algérie.

E-mail address: Achache_m@yahoo.fr

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

PROCESS-ORIENTED METRICS FOR APPLICATION DEVELOPMENT OUTSOURCING. A PRACTITIONER'S APPROACH

D. RADOIU AND A. VAJDA

Abstract. The paper proposes a framework (Process-Oriented Metrics for Application Development Outsourcing – POMADO) which aims to provide numeric scores characterizing the performance of application development (AD) of the outsourcing process. The purpose of this framework is to obtain knowledge in order to gain increased insight in how to intervene in the outsourcing process. The research is aimed at both the scientific and practicing communities, giving two purposes. (1) Further the modeling of application development outsourcing process and its formal/quantitative characterization; (2) Provide practical guidelines on how to design metrics that become useful for quantitatively managing AD outsourcing process. Data on which the proposed metrics are based have been acquired through informal discussions with project managers from five companies involved in software development outsourcing, post delivery assessment of project results and available literature.

Keywords: Process Metrics, Measurement, Outsourcing Process Management, Application Development, Improvement.

1. Introduction

One of the most important issues in outsourcing process is assessing the quality of the process, both qualitative and quantitative. The two stakeholders (Partner and Vendor) involved in the outsourcing process usually have different views and expectations.

The basic idea of this research is to identify meaningful measurements of the outsourcing process so that both major stakeholders (Partner and Vendor) pinpoint problem areas and take converging corrective actions if needed or take action to improve the process.

The biggest challenge in establishing an effective metrics program in the outsourcing process has on one side to do with the formulas, statistics, and analysis but the real difficulty lies in determining which metrics are valuable to both organizations involved in the process, and which procedures are most efficient for using these metrics.

Data on which this research is based was provided by Infopulse, an organization of five companies specialized in AD and outsourcing services. Infopulse has successfully implemented an outsourcing process metrics program and provides

Received by the editors: 1/10/2004.

quality outsourcing services to several EU based companies (i.e. the collection, interpretation, distribution, and usage to optimize cooperation). Although the limited number of companies and available data represents a limitation of the research, we are strongly encouraged by the organization's success.

The research was organized as a practical project at Infopulse and a research project at Petru Maior University of Tirgu Mures.

2. Methodology

The steps we considered in this research are [7]:

(1) Stating the goals of the metrics (e.g. assessing the quality of a request for proposal, assessing the effectiveness of communication, use of statistical data for planning);
(2) Detailing a clear model of the AD outsourcing process;
(3) Identifying the meaningful elements to be measured (directly or indirectly);
(4) Analyze the measurements using a Performance Analysis Model;
(5) Validate the metrics.

The present paper addresses the first three steps.

3. Goals of POMADO

The goal of the proposed framework is to quantitatively manage the AD outsourcing process to achieve the established quality and outsourcing process-performance objectives.

Although the overall process performance is characterized by both outsourcing process measures and delivered product measures the focus of POMADO is on the process.

The benefits of POMADO are:

- Helps the stakeholders (Vendor and Partner, respectively supplier organization and outsourcing organization) predict whether the outsourcing project will be able to achieve its quality and process-performance objectives;
- Helps the project managers understand the nature and extent of the variation experienced in the outsourcing process performance, and recognizing when the project's actual performance may not be adequate to achieve the project's quality and process-performance objectives;
- Determine whether the outsourcing processes are behaving consistently or have stable trends (i.e., are predictable);
- Aware that Vendor and Partner have sometimes conflicting interests (e.g. time related, financial related, human resource related) POMADO focuses only on metrics acceptable by both parties to quantitatively characterize the outsourcing process.

4. The AD Outsourcing Process Model

The process model is depicted in Picture 1 [2]

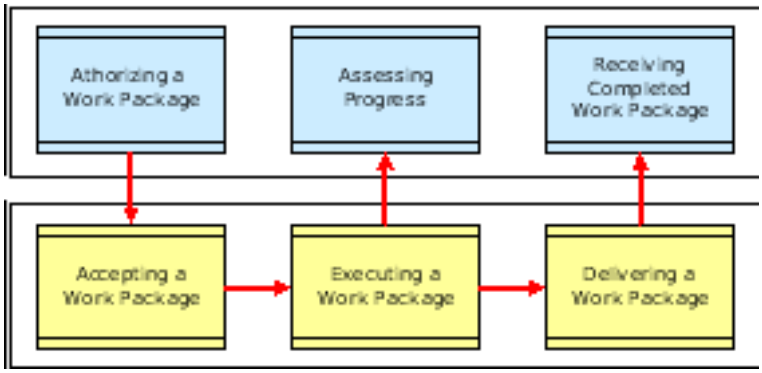

FIGURE 1

The upper level relates to Partner workflow, the lower level relates to Vendor workflow and the arrows depict the complex interfaces between phases within Partner's or Vendor's organization or the complex interaction between Partner and Vendor (e.g. syncron/asyncron exchange of information, change requests, etc.).

Executing a work package in AD is largely covered by different models which could be roughly described as: waterfall, incremental and iterative AD models.

The Waterfall Model (Picture 2) is mostly used when the requirements are known from the beginning and are not changed very often [3].

The Incremental Model (Picture 3) is reducing risks by incrementally assessing progress and taking corrective actions, increasing the success rate for medium to large projects.

The Iterative Model (Picture 4) is associated with large projects, changing requirements, incremental deliveries.

Although there is no consensus for such a classification, for practical purposes, the size (time and effort) of a project is usually described as: large, medium, small and very small (Picture 5) and different AD models are recommended (Table 1) [4].

Project Size	Waterfall	Iterative	Incremental
Large	If duration > 3 months, splitting into increments / iterations suggested	√	√
Medium	√	√	Not recommended
Small	√	Not recommended	Not recommended
Very Small	√	Not recommended	Not recommended

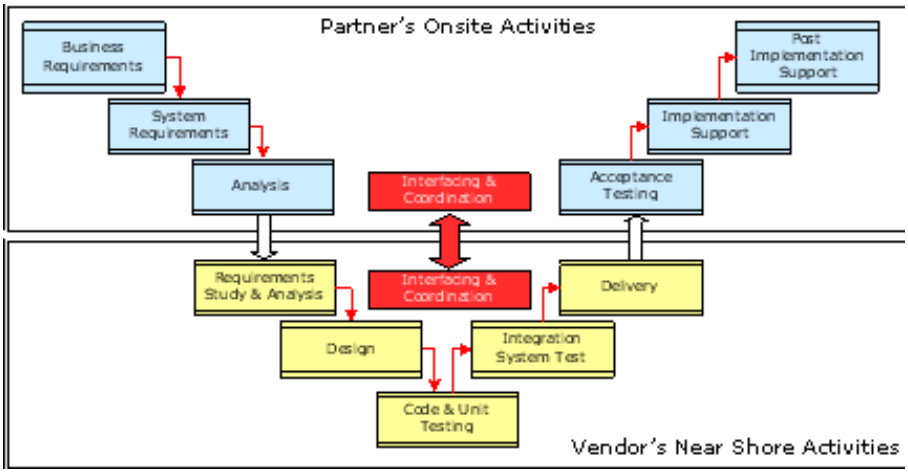

FIGURE 2

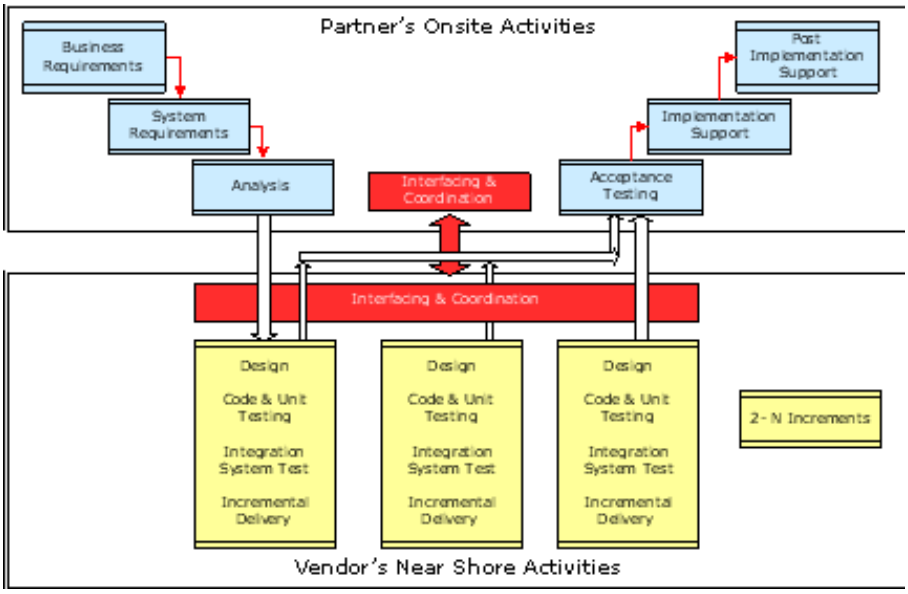

FIGURE 3

Although Infopulse Group was involved in large projects spanning over several years, availability of consistent data restricted our research to very-small, small and medium size projects.

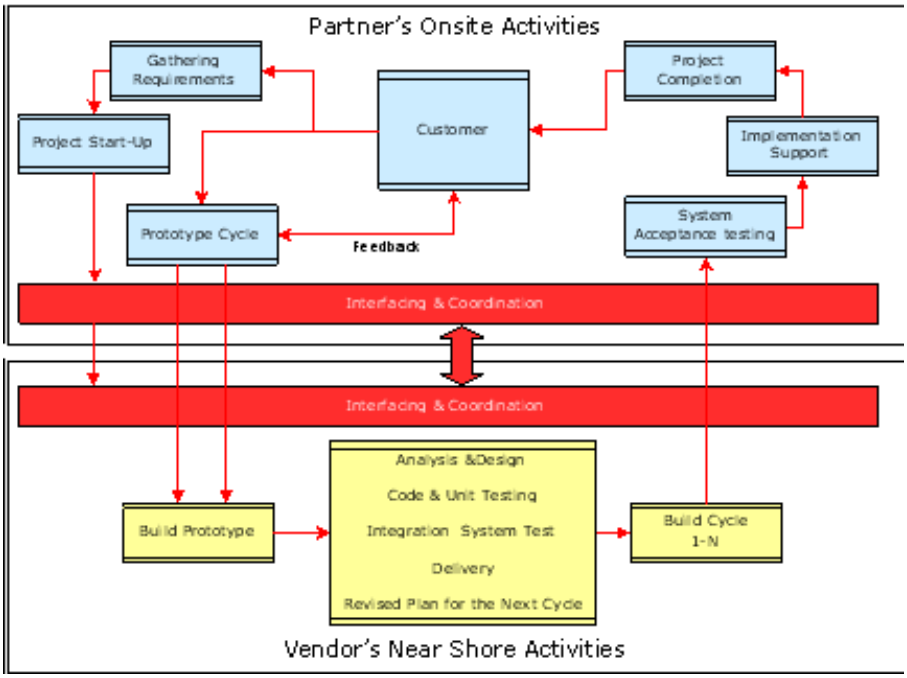

FIGURE 4

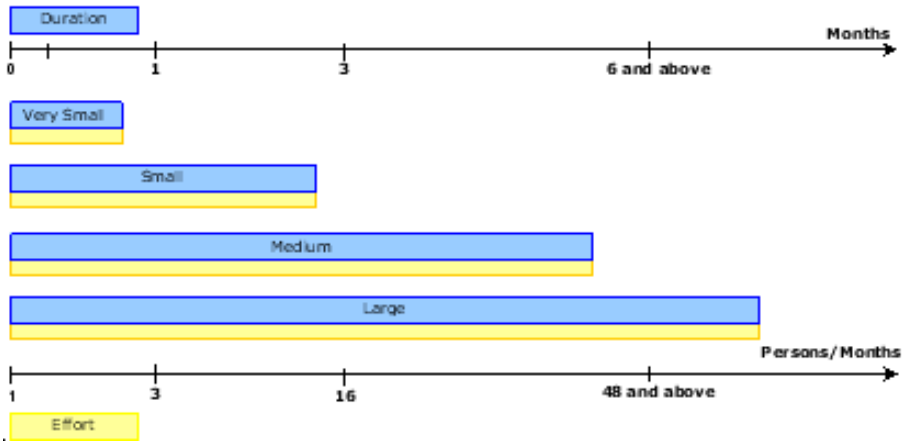

FIGURE 5

5. POMADO Direct Measurements

POMADO first aim is to provide numeric scores to characterize the performance of application development (AD) of the outsourcing process. Unlike AD development within organization, outsourcing means two stakeholders (Vendor and Partner) with partly conflicting goals. Both direct and indirect measurements proposed by the framework are focused on improving Partner-Vendor co-operation by pinpointing problem areas so that remedies can be developed and the outsourcing process can be improved. Therefore the criteria used in selecting metrics relate only to usefulness for both parties in managing and improving AD outsourcing process.

The criteria we considered are:

- Collected data must be relevant for the interaction between Partner and Vendor;
- Collected data must cover the interaction between Partner and Vendor;
- Collected data should be validated by Partner and Vendor;
- Collection process should be economic (i.e. maximum benefits/minimum effort);
- Partner and Vendor although with different views on the process must agree on the interpretation of metrics (before the collection) for this will determine the relevant areas for joint intervention (e.g. Communication between the two organizations);
- A key area in the outsourcing process (near-shore or off-shore delivery model) is communication (synchron/asynchron).

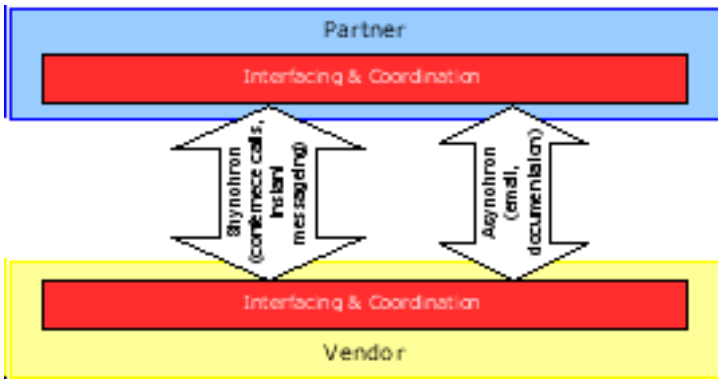

Figure 6

Asynchron communication always produces artifacts (i.e. documents, e-mails). By synchron communication we understand conference calls (video, phone), voice over IP and even Instant Messaging. POMADO direct measures are depicted in Table 2.

Direct Measures	Units	ID	Description
Definition (RFP, PDP, BR, etc)	LOD (Line of documentation)	O1	Refers to the number of lines needed for Work Package definition. E.g. Request for Proposal (RFP) Project Development Proposal (PDP) Business Requirements (BR) Functional Requirements (FR) Risk Assessment (RA) Object Model (OM)
Status Report	Nr of reports/month	O2	The frequency of the Status report indicates the transparency of the project.
Change Requests	Nr of requests	O3	The number of functional point changes requested by the partner or vendor
Early Change requests	Nr of requests	O4	The number of changes requested during the project analyzes.
Questions & Answers	Nr of questions & Answers	O5	Represents the number of questions necessary to understand, clarify the requirements or proposals
Response time	Minutes	O6	Represents the elapsed time between posting a question and receiving the answer
E-mails	LOE (Lines of emails),	O7	Represents the amount of information sent trough email
	Nr of emails	O8	The number of the emails sent.
Response time	Minutes	O10	Represents the elapsed time between sending an email with questions and receiving the answer. This is important also for the emails marked as High importance.
Conference calls	Minutes	O11	The conference call can be a phone-, a video-, a voice over IP conference. The duration of the conference is measured.
Instant messaging	Minutes	O12	Duration of the communication
Project Size	MD (ManDays, ManHours)	O13	Project Duration
Functional Points (FP)	Number	O14	The number of Functional points of the Work Package

6. POMADO Indirect Measures

INPUT (AUTHORIZING & ACCEPTING)
Work Package Accuracy
Work Package Stability

Work Package Definition Level (LOD/FP)
AD CYCLE (EXECUTING & ASSESING)
Average Delivery Time
Change Request Density
Q&A Density
Average Response Time
OUTPUT (DELIVERING & RECEIVING)
Acceptance Criteria Definition

Work Package Accuracy

This metrics characterizes Partner's project documentation. An ideal documentation (WPA = 1) requests no clarification questions (assuming there's business knowledge and technical knowledge at the Vendor's end). A less ideal documentation means WPA approaching zero or even worst, taking negative values for a poor documentation.

$$\mathrm{WPA} = 1 - \frac{\text{questions}}{\mathrm{FP}}$$

Work Package Stability

This metrics characterizes the quality of the project definition: correctness of the requested functionalities, understanding client business requirements, quality of the analysis, design and RFP. Ideal stability means that no change requests are made after the work package has been accepted by the Vendor.

$$\mathrm{WPS} = 1 - \frac{\text{change requests}}{\mathrm{FP}}$$

Work Package Definition

$$\mathrm{WPD} = \frac{\mathrm{LOD}}{\mathrm{FP}}$$

Average Delivery Time

ADT represents the time elapsed from meeting entry criteria to meeting exit criteria for a Functional Point

$$\mathrm{FPAT} = \frac{\dfrac{\text{project size}}{\text{resource number}}}{\mathrm{FP}}$$

Change Request Density

$$\mathrm{CRD} = 1 - \frac{\text{change requests during execution}}{\mathrm{FP}}$$

Q&A Density

$$\mathrm{QAD} = \frac{\text{questions}}{\mathrm{LOD}}$$

Average Response Time

$$\text{RAT} = \frac{\sum \text{response time}}{\text{questions}}$$

7. Further study

Many companies have implemented metrics programs to support the managers in their decision yet nearly 80% of software metrics programs fail within the first two years [1]. The reasons range from difficulties to collect reliable and useful data to the lack of a Performance Analysis Model to provide answers to product, process or overall project related questions.

The success of a Vendor-Partner relation relies both on managing the process and improving it. It implies agreement of how to quantitatively answer questions like:

- How large is the process interference generated by Change Requests?
- Is the Partner's response time compatible with project constraints?

To consistently answer such questions a performance analysis model - based on interdependencies of project data - is required. This provides the first of the two fundamental characteristics [5], a sound conceptual, theoretical basis. The second one is a statistically significant validation.

Acknowledgments

We would like to thank Infopulse Group for supporting this research and Infopulse QA team for reviewing the paper and providing insightful suggestions.

References

[1] Dekkers, C.S., The Secrets of Highly Successful Measurement Programs, Cutter IT Journal, vol 12 no. 4, pp. 29-35
[2] PRINCE 2, http://www.prince2.com/
[3] Pressman R. S., Software Engineering, A Practitioner's Approach, McGraw-Hill, 1982, pp. 24-25
[4] * * *, Project Management Guidelines, Intranet, Infopulse Group, 2004
[5] Mills E. E., Software Metrics, SEI Curriculum Module SEI-CM-12-1.1, 1988, pp.6-7
[6] Goethert W, Brad C., Managing Software Projects with Metrics, Software Engineering Institute, Carnegie Mellon University, SEI Symposium Conference, 2000
[7] Kirchner P. "Measurements and Management Decisions," in Measurement: Definitions and Theories, C. West Churchman & Philburn Ratoosh, ed. New York, N.Y.: John Wiley & Sons, Inc., 1959.

Department of Mathematics and Computer Science, Petru Maior University, Targu Mures, Romania

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

RMI VERSUS CORBA: A MESSAGE TRANSFER SPEED COMPARISON

FLORIAN MIRCEA BOIAN AND RAREŞ FLORIN BOIAN

Abstract. RMI (Remote Method Invocation) [5][1] and CORBA (Common Object Request Broker Architecture) [9][1] are two technologies for communicating between objects distributed across a network. The use of distributed objects is an attractive paradigm for designing distributed applications because of the simple abstraction layer that hides the network communication details. This article presents the results of tests that compared the speed of the message transfer between applications using these technologies. Five of the various implementations available for RMI and CORBA have been selected for the purpose of these tests. The interoperability between these implementations is also discussed.

1. Introduction

Distributed applications are very common in today's world. Every serious business has a web site that besides containing static information often provides automatically generated pages. Intranet applications within the domain of the same company usually provide direct communication between desktop applications and the server. The very popular file sharing P2P systems have to bring together computers from all over the world. In such environments, a distributed application has to be ready to work on various platforms (Intel, SPARC, Alpha, etc.) and operating systems (Windows, Solaris, Linux, etc.). More than that, such applications have to interface with legacy systems written in various programming languages. Although socket communication simplifies the communication in heterogeneous environments, the complexity of the communication protocols is ever increasing.

In 1989, the Object management Group (OMG) released the first specification of CORBA [9][1], a standard that allowed applications distributed in heterogeneous environments to exchange objects. The design made transparent to the developer the platforms on which the communicating applications were running as well as the programming language in which they were implemented.

Approximately at the same time, Microsoft created a similar product called Distributed Object Model (DOM) implemented in C++. The main restriction brought by DOM was the lack of implementations on non-Microsoft platforms, which made it unsuitable for heterogeneous network environments running other systems than Windows.

After the Java programming language was widely adopted as a programming language of choice, Sun Microsystems introduced in 1995 the Remote Method Invocation (RMI) [5][1] technology. RMI allowed Java objects owned by different

Received by the editors: 1/10/2004.

applications to call each other's methods and receive back the results of the process. The RMI benefited from Java's platform independence and portability, but did not adhere to the CORBA standard, thus it did not provide means to connect to legacy systems implemented in languages other than Java. The initial RMI implementation relied on URL naming to access remote objects. The Java Naming and Directory Service (JNDI) [2][8] brought to RMI a flexible and logical modality of naming services, beside the existing direct addressing approach [2][9].

Besides RMI, Sun Co. packaged with the release of Java 2, a native implementation of the CORBA standard, named Java IDL. With a few exceptions concerning internationalization [10], the Java IDL implementaion instrumented the Java distribution with full capabilities to communicate with system implemented in other programming languages.

To close the gap between the two supported standards (RMI and CORBA) and to bring the RMI ease of use into cross-language programming, Sun Co. released the RMI-IIOP solution. Using RMI-IIOP, RMI servers can communicate with CORBA clients implemented in any language, by respecting to a few restrictions [11]. The RMI-IIOP is basically built on top of Java IDL the rmic compiler (given the –iiop command line option) being able to generate IIOP stubs, ties and IDL.

Beside the four Java-based technologies mentioned above, the free omniORB [4] C++ implementation of CORBA was used to test the interoperability of CORBA and the performance difference among different implementation languages.

Choosing between using CORBA and using RMI has been a problem ever since Sun Co. decided to support both these competing technologies. Both options have advantages and disadvantges related to the ease of use, flexibility and level of integration with the Java language [12]. A study of their inter-communication performance may help taking a decisionin this matter.

2. Experimental setup

The five technologies used in this experiment are: classical RMI, classical RMI using JNDI as name service, RMI with IIOP, Java IDL and omniORB. For performance measurement purposes a simple Echo client-server application was developed. The clients send strings to the server, and the server objects transform the message to upper case letters and prefix it with the server request processing time measured in milliseconds . The resulting string is returned to the client. The same program was implemented for all five technologies. Figure 1 shows the communication diagram of the experimental setup.

Each of the five servers creates its context and then creates an EchoImpl object specific to its implementation, as seen in Figure 1. Each object is then registered (bind) to a name service. The RMI and RMI over JNDI use the standard rmiregistry name service. The remaining three implementations use tnameserv as name service, which is a default name service provided with the JavaIDL package. The omniORB implementation offers its own name service implementation (omniNames) but it can only be used by omniORB applications. After the EchoImpl object is registered, the servers stay idle waiting for requests.

The four Java clients are all implemented as standalone applications. The omniORB client is implemented as a WIN32 console application. Each program is given on the command line the name of the remote object to invoke and the string to send. Each client sends the same message a predefined number of times to the server, records timing information and then stops. The time is measured

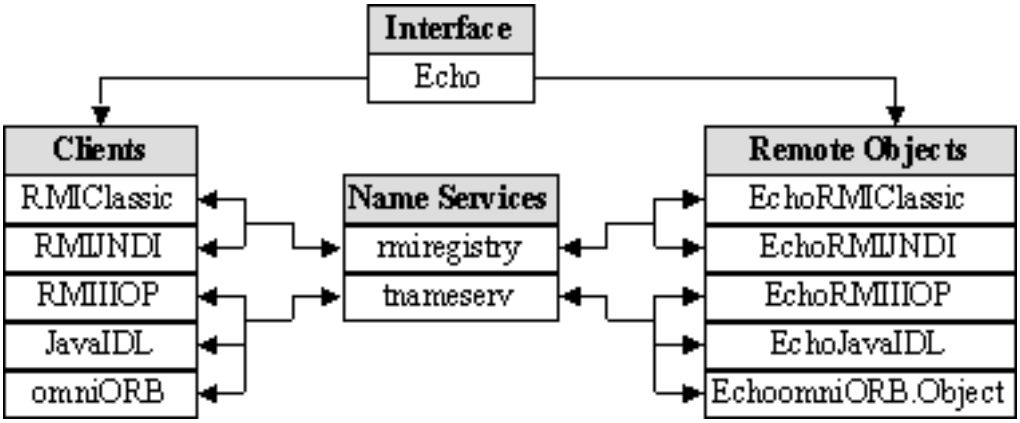

FIGURE 1. Architecture of the experiment

around the remote object invocation. For accuracy, the server-side processing time returned with the response is subtracted from the total time elapsed during the call. This approach eliminates the program startup time from the measured time and gives a more accurate performance measurement smoothing out inherent delays caused by the operating system by averaging across all the readings. Neither the clients, nor the servers write anything to the standard output (console) to avoid affecting the communication time measurement. In addition to measuring the performance of the remote method calls, the time necessary for naming lookups were also measured. The same multiple iteration approach was used as in the case of the remote invocations.

The tests were executed on an AMD Athlon 1.4 MHz computer running Windows 2000 Professional. To eliminate any delays caused by network traffic, the servers and the clients resided on the same machine. The servers were started and continued to function while the clients were executing one at a time.

3. IMPLEMENTATION

The interface necessary to implement the remote objects invoked by the client applications is shown in the table below. Two implementations are necessary, one using IDL for the CORBA technologies and another one using Java for the pure RMII applications.

IDL	Java

interface Echo {
  string echoString(in string s);
}; //Echo
	import java.rmi.*;
	public interface Echo extends Remote {
	  public String echoString(String s)
	  throws RemoteException;
	}//Echo

The IDL and Java interface implementations are very similar. Both define an echoString method that receives and input parameter of type string and returns a result of type string.

A generic Java implementation of the server object is presented below. The implementation differences between the four Java technologies consist only in the class declaration, namely in class extended and the interfaces that are implemented. The values of <EXTENDS> and <IMPLEMENTS> for each of the four Java based technologies are given in the table following the source code.

```
public class EchoImpl <EXTENDS> <IMPLEMENTS> {
   int invoke;
   public EchoImpl() throws RemoteException {
      super();
      invoke = 0;
   }//EchoImpl.EchoImpl

   public String echoString(String s) {
      invoke++;
      return invoke+"\t"+s.toUpperCase();
   }//EchoImpl.echo
}//EchoImpl
```

Technology	<EXTENDS >	<IMPLEMENTS >
Classical RMII	extends UnicastRemoteObject	implements Echo
RMI with JNDI		
RMI-IIOP	extends PortableRemoteObject	
Java IDL	extends EchoPOA	

Similar to Java, the C++ omniORB implementation is presented below.

4. Interoperability between the tested technologies

There are twenty-five possible ways to select a client and a server implementation from the five chosen technologies, but not all of them are interoperable. The ten combinations that are compatible with each other are presented in the table below.

As implementations of the CORBA standard, Java IDL, omniORB and RMI-IIOP are expected to be fully compatible to each other. Indeed, the first two technologies can be used on either the server or the client end of the application. However, their interaction with RMI-IIOP requires additional steps.

The communication between RMI-IIOP and Java IDL is conditioned by having access to the other end's technology stub. In other words, the variable CLASSPATH in the client's environment has to contain the class stub of the server object, and reverse.

```
class EchoImpl : public POA_Echo, public PortableServer::RefCountServantBase {

   private:
      int invoke;
   public:
       EchoImpl() {
            invoke = 0;
      }//EchoImpl.EchoImpl

      char* echoString(const char* s) {
         int i;
         char t[1000];
         invoke++;
         sprintf(t, "%d \t %s", invoke, s);
         _strupr(t);
         return CORBA::string_dup(t);
      }//EchoImpl.echoString
};//EchoImpl
```

Client \ Server	Java IDL	omniORB	RMI-IIOP	Classic RMI	RMI-JNDI
Java IDL	X	X	X		
omniORB	X	X			
RMI-IIOP	X	X	X		
Classic RMI				X	
RMI-JNDI					X

An intermediate level of Java IDL implementation is necessary in order to communicate between RMI-IIOP and omniORB, and in general with other non-Java CORBA applications.

The pure RMI implementations, with or without using JNDI cannot communicate outside their technology bounds. The communication between pure classic RMI and pure RMI-JNDI is made impossible by the naming protocol used. In order to access an object offered by an RMI-JNDI server, a classic RMI client needs to access the JNDI service to access the object. The reverse applies to an RMI-JNDI application trying to access an object offered by a classic RMI application.

The similar with RMI classic, RMI over JNDI can connect only with RMI over JNDI partner.

5. Results of the first experiment

The test client and server programs were executed several times with different iteration counts for name lookup operations and remote method calls. The iteration numbers ranged from 1 to $6x10^5$ in multiples of powers of ten. The results presented below are chosen from the run with most repetitions. Three aspects of the results are discussed below: variations in the measurements, the name lookup duration, and the duration of a remote call.

5.1. **Measurement Variations.** The measurements focused on the duration of the operations. Running the programs with iteration numbers different by orders of ten, made visible that initial operations are slower than the subsequent ones. The two tables below present the average duration of lookup and remote call operations measured over various numbers of repetitions. The configuration chosen is classic

RMI client and server but the behavior is consistent for all the other nine situations. The durations are expressed in milliseconds.

Lookup Iteration Count	Lookup Total Time	Lookup Average Duration
1	210	210.00
11	240	21.81
111	581	5.23
1111	2997	2.69
11111	13922	1.25
111111	123044	1.10

Method Call Iteration Count	Method Call Total Time	Method Call Average Duration
1	10	10.00
20	30	1.50
300	320	1.06
4000	2201	0.55
50000	15310	0.30
600000	169286	0.28

For both, name lookup operations and remote method call operations, it is visible that the first operations are significantly slower than subsequent ones. For name lookups, the operation duration is around one millisecond in for large number of iterations. Considering the second case with 11 iterations, if the duration of the first operation (210 ms measured on the line above) is subtracted from the total duration and the average of the remaining calls is averaged we get about 3 ms per call. This is still larger than the 1 ms obtained in the last case. Repeating this operation for the remaining rows, the lookup durations decrease slowly toward 1 ms, which proves that although the initial operations are by far the slowest, there are further interferences (possibly external to the test application) that slow down the communication. The same conclusion can be reached examining the remote method call measurements.

5.2. **Name Lookup Duration.** The table below presents the duration of name lookup measurements for the configuration with 111111 interations. It is apparent that the lookup operation durations are dependent on the client technology. The configuration involving omniORB are consistently faster than the rest. The RMI implementations are faster the the Java CORBA-compatible ones. This is most likely due to the lighter RMI implementation and the direct integration with the Java language, with the need for conversion in a universal format.

5.3. **Remote Method Call Duration.** The table below presents the duration of remote method call measurements for the configuration with 60000 interations.

The duration of the remote method calls are reversed in behavior when compared with those of the name lookup operations. This time the executionm times are dependant on the server technololgy instead of the client technology. The configurations with omniORB as server are significantly fatser then the rest, while those with omniORB as client are faster than thier counterparts. The higher speed

Client Technology	Server Technology	Avg. Duration per Lookup
Java IDL	Java IDL	1.418329
Java IDL	omniORB	1.354654
Java IDL	RMI-IIOP	1.434691
omniORB	omniORB	0.929962
omniORB	Java IDL	1.003096
RMI-IIOP	RMI-IIOP	2.515178
RMI-IIOP	omniORB	1.880749
RMI-IIOP	Java IDL	2.658926
Classic RMI	Classic RMI	1.107397
RMI-JNDI	RMI-JNDI	1.205047

Client Technology	Server Technology	Avg. Duration per Call (client)	Avg. Duration per Call (server)
Java IDL	Java IDL	0.997540	0.011038
Java IDL	omniORB	0.308578	0.003203
Java IDL	RMI-IIOP	1.004380	0.009730
omniORB	omniORB	0.161997	0.001988
omniORB	Java IDL	0.690818	0.009608
RMI-IIOP	RMI-IIOP	1.033056	0.010533
RMI-IIOP	omniORB	0.273586	0.002858
RMI-IIOP	Java IDL	0.988003	0.011101
Classic RMI	Classic RMI	0.282143	0.006160
RMI-JNDI	RMIJNDI	0.305676	0.007006

of the omniORB implementation is obviously explained by the fact that it is implemented in C++ and hence the program is a native and optimized executable instead of interpreted bytecode, which is the case of Java.

Another interesting aspect of the results is in the duration of teh server side method execution. Although the EchoImpl object executes the same code in all cases, the duration times differ amongst the RMI and and Java/CORBA implementations.

6. Conclusions

The experiments presented above were intended to assist in deciding the technology to use when starting a project. The results favor the C++ implementations which is expected given the optimized native code versus the Java bytecode. However, these tests cover only a small aspect of the many issues to be taken into account when choosing a technology. In addition to application speed, the following important criteria have to be considered before making a decision.

Licensing costs. While Java distributions are free, most of the C++ implementations are not.

Development team experience. Developing in a familiar environment will always yield faster and better results than using a new platform that first needs to be learned.

Flexibility. If the distributed application includes Java components, using RMI can make a developer's job easier with features such as distributed garbage collection, object passing by value, or dynamic class downloading [10].

Platform independence. If the application has to run in a heterogeneous environment, Java is a better option that saves the effort of adapting the implementation to each platform separately.

Productivity. Although the familiarity of the development team with one technology or another will be the decisive factor in this matter, it is generally considered that it is more productive to program in Java than in C++. This is mostly due to the lack of memory handling issues, hidden by the automatic garbage collector, and the consistent and simple design of the Java library.

References

[1] Harkey O. Client – server programming with Java and Corba, John Wiley, 1999
[2] Todd N., Szolkowski M. Locating Resources Usind JNDI (Java Naming and Directory Interfaces, SAMS, www.developer.com/java/ent/article.php/2215571
[3] Vinoski S. CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments. IEEE Communication Machine, 35, 2, 1997. http://www.iona.com
[4] * * * AT&T Cambridge Laboratory http://www.uk.research.att.com/omniORB or omniORB.sourceforge.net.
[5] * * * Java^{TM} Remote Method Invocation Documentation, http://java.sun.com/docs/guide/rmi/index.html
[6] * * * The JNDI Tutorial: Building Directory-Enabled Java Applications http://www.java.sun.com/products/jndi/tutorial/
[7] T. Suganuma, T. Ogasawara, M. Takeuchi, T. Yasue, M. Kawahito, K. Ishizaki, H. Komatsu, T. Nakatani, Overview of the IBM Java Just In Time Compiler, IBM SYSTEMS JOURNAL, Vol 39, No 1, 2000, http://www.research.ibm.com/journal/sj/391/suganuma.pdf
[8] * * *, Java^{TM} 2 SDK, Standard Edition Documentation, http://java.sun.com/j2se/1.4.2/docs/index.html
[9] * * * Object Management Group *CORBA Services Specification*, http://www.omg.org/library/csindx.html
[10] Java 2 RMI and IDL Comparison, http://lisa.uni-mb.si/∼juric/J2rmiidlc.pdf
[11] RMI-IIOP Programmer's Guide, http://java.sun.com/j2se/1.4.2/docs/guide/rmi-iiop/rmi_iiop_pg.html
[12] Java RMI & CORBA A comparison of two competing technologies, http://www.javacoffeebreak.com/articles/rmi_corba/

Babes Bolyai University, Cluj Napoca,Romania
E-mail address: florin@cs.ubbcluj.ro

Rutgers University, New York
E-mail address: boian@caip.rutgers.edu

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

UNBOUNDED AND BOUNDED PARALLELISM IN BMF. CASE-STUDY: RANK SORTING

VIRGINIA NICULESCU

Abstract. BMF is a formalism that allows us to design parallel programs independently of the target architecture, and to transform the programs into more efficient programs using equational reasoning. We show in this paper that even the abstractness of BMF is very high, bounded and unbounded parallelism can be expressed in this model, and also that BMF allows us to transform a program into different variants, each of them being more appropriate for a specific architecture type. We consider the case-study of rank sorting, for which we construct first a general unbounded parallel program. Then, we transform the program for bounded parallelism, by imposing a limited number of processors. Three variants are obtained. The implementations of the programs onto shared memory, distributed memory, and pipeline architectures are analyzed, too.

1. Introduction

The problem of constructing parallel software is much more difficult than for the sequential case. This is ultimately because there is no longer a single computation model to play the role that the von Neumann model plays in sequential computing. The goals of parallel software construction are also more ambitious. To be useful over any significant period of time, a parallel program must be able to be executed by many parallel computers, differing internally in significant ways. Thus the programmer's problem is not only to build an optimal program for a particular parallel computer, but to build one that is optimal for many different parallel computers [6].

2. Bird-Meertens Formalism

Bird-Meertens formalism (BMF) on lists was originally created for the design of sequential programs [1], but it becames very popular in the parallel setting. In BMF, higher-order functions (functionals) capture, in an architecture-independent

Received by the editors: September 2004.

2000 *Mathematics Subject Classification.* code, code.

1998 *CR Categories and Descriptors.* code [**Topic**]: Subtopic – *Detail*; code [**Topic**]: Subtopic – *Detail* .

way, general idioms of parallel programming, which can be composed for representing algorithms. BMF functionals use elementary operators and functions as parameters, so that a BMF expression represents a class of programs which can be reasoned about, either taking into account particular properties of the customizing functions or independently of them [5, 2, 3].

2.1. **BMF Notation.** The basic data structure it is considered the non-empty list: $[\alpha]$, where α is the elements type. Function application is denoted by juxtaposition, considering the tightest binding and association to the left.

The simplest functional of BMF is *map*, which applies a unary function f, defined on the elements type, to each element of a list:

$$map\ f\ [x_1, x_2, \ldots, x_n] = [f\ x_1, f\ x_2, \ldots, f\ x_n] \tag{1}$$

The functional *map* is highly parallel since the computation of f on different elements of the list can be done independently if enough processors are available. The elements of a list can be lists again and f may be quite a complex function or composition of functions.

In addition to the parallelism of *map*, BMF allows to describe tree-like parallelism. This is expressed by the functional *red* (for reduction) with a binary associative operator $\oplus$:

$$red\ (\oplus)\ [x_1, x_2, \ldots, x_n] = x_1 \oplus x_2 \oplus \cdots \oplus x_n \tag{2}$$

Reduction can be computed on a binary tree with $\oplus$ in the nodes. The time of parallel computation depends on the depth of the tree, which is $\log n$ for an argument list of length n.

Other functionals may be defined, but these two are the most important and the most used.

Functions are composed in BMF by means of functional composition $\circ$, such that $(f \circ g)\ x = f\ (g\ x)$. Functional composition is associative and represents the sequential execution order.

BMF expressions - and, therefore, also the programs specified by them - can be manipulated by applying semantically sound rules of the formalism. Thus, we can employ BMF for formally reasoning about parallel programs in the design process. BMF is a framework that facilitates transformations of programs into each others. A simple example of BMF transformation is the map fusion law:

$$map\ (f \circ g) = map\ f \circ map\ g \tag{3}$$

If the sequential composition of two parallel steps on the right-hand side of Eq. (3) is implemented via a synchronization barrier, which is often the case, then the left-hand side is more efficient and should be preferred.

A very important skeleton in BMF is the homomorphism that reflects the divide-and-conquer algorithms [2, 3]. Still, we will not refer to it in this paper.

We are going to focus on how BMF expresses unbounded and bounded parallelism.

3. Bounded and Unbounded Parallelism

For serial algorithms, the time complexity is expressed as a function of n – the problem size. The time complexity of a parallel algorithm depends on the type of computational model being used as well as on the number of available processors. Therefore, when giving the time complexity of a parallel algorithm it is important to give the maximum number of processors used by the algorithm as a function of the problem size. This is referred to as the algorithm's *processor complexity*. For example, a serial algorithm to find the maximum of a set with n elements has complexity $O(n)$, since it requires $n-1$ comparisons. In contrast, a trivial parallel algorithm for the same problem has time and processor complexities $O(\log n)$ and $O(n)$ respectively.

The synthesis and analysis of a parallel algorithm can be carried out under the assumption that the computational model consists of p processors only, where $p \geq 1$ is a fixed integer. This is referred to as *bounded parallelism*. In contrast, *unbounded parallelism* refers to the situation in which it is assumed that we have at our disposal an unlimited number of processors. Let us assume that a parallel algorithm A solves a problem of size n on p processors. If there exists a polynomial F such that for all n, $p < F(n)$, then the number of processors is said to be polynomially bounded; otherwise it is polynomially unbounded.

From a practical point of view algorithms for bounded parallelism are preferable. It is more realistic to assume that the number of processors available is limited. Although parallel algorithms for unbounded parallelism, in general, use a polynomially bounded number of processors(e.g. $O(n^2)$, $O(n^3)$, etc.) it may be that for very large problem sizes the processors requirement may become impractically large. However, algorithms for unbounded parallelism are of great theoretical interests, since they give limits for parallel computation and provide a deeper understanding of a problem's intrinsic complexity.

So, because of these, the right way for designing parallel programs is to start from unbounded parallelism and then transform the algorithm for bounded parallelism. There are two methods for carrying out such transformations. One is the decomposition of the problem into subproblems of smaller sizes, and another is the decomposition of the algorithm into substeps in such a way that each of them can be executed using a smaller number of processors.

The BMF programs usually reflect the unbounded parallelism. The functional *map*, for example, has the time complexity equal to $O(1)$ with a processor complexity n (n is the list's length).

In order to transform BMF programs for bounded parallelism, we may introduce the type $[\alpha]_p$ of lists of length p, and affix functions defined on such lists with the subscript p, e.g. map_p [3]. Partitioning of an arbitrary list into p sublists, called *blocks* may be done by the *distribution* function:

$$dist^{(p)} : [\alpha] \to [[\alpha]]_p \tag{4}$$

The following obvious equality relates distribution with its inverse, flattening: $red(|) \circ dist^{(p)} = id$, where $|$ means concatenation.

Using these, we obtain the following equality for the functional *map*:

$$map\ f\ = flat \circ map_p\ (map\ f) \circ dist^{(p)} \tag{5}$$

where the function $flat : [[\alpha]]_p \to [\alpha]$ transforms a list of sublists into a list by using concatenation ($flat = red(|)$).

For reduction, the transformation for bounded parallelism is as it follows:

$$red(\oplus)\ = red_p(\oplus) \circ map_p\ (red(\oplus)) \circ dist^{(p)} \tag{6}$$

We will consider that only the functions with subscription p will be distributed over the processors. The others will be sequentially computed.

4. Case-study: Rank Sort

The idea of the rank sort algorithm is the following: determine the rank of each element in the unsorted sequence and place it in the position according to its rank. The rank of an element is equal to the number of elements smaller than it [4].

Rank sort is not exactly a good sequential sorting algorithm because the time complexity in the sequential case is: $O(n^2)$ (n is the length of the sequence). But this algorithm leads to good parallel algorithms.

Using this simple example, we are going to illustrate that the abstract design using BMF can comprise different cases that may appear at the implementation phase: shared memory versus distributed memory and pipeline computation, and different numbers of processors. So, the abstractness of this formalism does not exclude performance.

4.1. **BMF design.** By using the definition of the method we arrive to the following simple BMF program:

$$\begin{array}{l} rank : [\alpha] \to [\alpha] \\ rank\ l = map\ (count\ l)\ l \\ \\ count : [\alpha] \times \alpha \to [\alpha] \\ count\ l\ x = red(+) \circ map\ (f\ x)\ l \\ \\ f : \alpha \times \alpha \to \alpha \\ f\ x\ y = \begin{cases} 1, \text{ if } x \geq y \\ 0, \text{ if } x < y \end{cases} \end{array} \tag{7}$$

A simple and obvious transformation can be made:

$$red(+) \circ map\ (f\ x)\ l = red(+)\ (map\ (f\ x)\ l) \tag{8}$$

which simple means that the application of the function $f\ x$ is made in the same step with the reduction, not in a separate sequent step.

4.2. **Unbounded parallelism.** We will analyze first the unbounded parallelism. For computing the first functional *map* we need a number of processors equal to the length of the input list – n. Each application of the function *count* l is a reduction which can be computed with $O(\log n)$ time complexity and $O(n)$ processor complexity. So, for whole program the unbounded time complexity is $O(\log n)$ with the processor complexity equal to $O(n^2)$.

4.3. **Bounded parallelism.** For bounded parallelism we need to transform the program by imposing the number of processors to be equal to p. For the transformation we use the function $dist^{(p)}$.

As it may be noticed from the specification, the algorithm contains two phases: one represented by the function *map* and the other represented by the function *count*, each of them having the list l as argument. The function $dist^{(p)}$ simple divided the argument list into p balanced sublists. So we may apply it for the computation of the function *map*, or for the computation of the function *count*, or for both.

There are two cases that we have to take into account:

(1) $p \leq n$,
(2) $n < p \leq n^2$

4.3.1. *Case* $p \leq n$. If the number of processes is less or equal to the size of the sequence, than the function $dist^{(p)}$ may be applied only once: for the function *map*, or for the function *count*.

In the first case we obtain the following BMF program, by using the equational rule (5):

$$\begin{array}{l} rank\ l = (flat \circ map_p\ (map\ count\ l) \circ dist^{(p)})\ l \\ count\ l\ x = red(+)\ (map\ (f\ x)\ l) \end{array} \tag{9}$$

This means that each processor sequentially computes the ranks for n/p elements.

If we applied the function *dist* to the function *count* we obtain the following BMF program:

$$\begin{array}{l} rank\ l = map\ (count\ l)\ l \\ count\ l\ x = red_p(+) \circ map_p\ (red(+)) \circ (map\ (f\ x)) \circ dist^{(p)}\ l \end{array} \tag{10}$$

Here we have used the equational rule (6). This program sequentially computes the ranks of the elements, but the the ranks are computed in parallel using p processors. For computing the rank of an element x, the processors compare x

```
forall (i = 0; i < p; i++) do in parallel
  for (k = 0; k < n/p; k++) do
    global_read(A[i * n/p + k], ak);
    r = 0;
    for (j = 0; j < n; j++) do { count the numbers less than ak}
      global_read(A[j], aj);
      if (aj < ak)  r++;  fi
    rof
    global_write(r, R[i * n/p + k]);
  rof
llarof
```

FIGURE 1. The SM program for $p \leq n$

with all their n/p local elements and compute local ranks; then the local ranks are added.

The two cases reflect different ways for algorithm decomposition in substeps, which can be computed with p processors. We consider that the functions with the subscript p will be computed in parallel on the p processors, and the functions without subscription will be computed sequentially. This leads to the following time complexities: (n^2/p) for the first case, and $(n^2/p + n \log p)$ for the second. Obvious the first is the best.

Implementations for $p \leq n$. On *Shared Memory* (SM) architectures the list l is shared by all processors, so normally we will choose the first case with the better time complexity. We may transform the correspondent BMF program into the PRAM-like program described in the Figure 4.3.1.

Each processor makes n^2/p readings and n/p writings from/in the shared memory. If we consider a CREW architecture the complexity is $(n^2/p+\alpha(n^2/p+n/p))$, where α is the unit time for shared memory access.

On a *Distributed Memory* (DM) architecture, the second alternative is better since we have the list distributed over the processors. An element is broadcasted to all the processors during the step that computes its rank - so there are n steps. Each processor compares the current received element with the local elements and computes a local rank. Local ranks are summed using a tree like computation that represents the reduction. A psudo-MPI program is described in the Figure 4.3.1.

The time complexity is given by the following expression: $n(n/p+\log p(1+\beta)+ b\beta)$, where β is the unit time for communication, and constant b reflects the time for broadcast.

Pipeline architectures may also be used for implementing this algorithm. If there are n processors, each processor has a local value $a[i]$. The elements of

$Rank(mypid)$:
  $for\ (i = 0; i < n; i++)\ do$
    $Bcast(A[i])$; { the root is the process that contains A[i] }
    $rl = ComputeLocalRank(A[i])$;
    $Reduce(rl, mypid)$;
  rof

FIGURE 2. The DM program for $p \leq n$

the list are piped into the pipeline, and the rank is updated in each processor by comparing the piped value with the local value. The current rank is also piped into the pipeline. If $p < n$ than each processor makes more comparisons at each step. The time complexity, for this implementation, is: $(n+p-1)(n/p+\beta)$, where β is the unit time for communication between two processors.

4.3.2. *Case* $n < p \leq n^2$. If we have more than n processors and $p = q * r$, we may use the function *dist* for *map* and also for *count* computations. So, we arrive to the following BMF program:

$$\begin{array}{l} rank\ l = (flat \circ map_q\ (map\ count\ l) \circ dist^{(q)})\ l \\ count\ l\ x = red_r(+) \circ map_r\ (red(+)) \circ (map\ (f\ x)) \circ dist^{(r)}\ l \end{array} \tag{11}$$

The time complexity for this case is $(n/q)(n/r + \log r)$.

On a SM architecture the program differs by the program presented in Figure 4.3.1 with the fact that for each element the function *count* is computed using a tree-like computation. Each processor makes $n^2/p + (n/q)\log r$ readings and $(n/q)\log r$ writings from/in the shared memory.

For a DM architecture the processes may be arranged on a $q \times r$ mesh, and the data distribution of the first line may be replicated on the other lines of processors. Each line computes the ranks of a sublist with n/q elements. The time complexity is $n^2/p + n/q(\log r(1+\beta) + b\beta)$.

A pipeline architecture with more than n processors is no useful for this problem.

5. CONCLUSIONS

We have analyzed here the developing of parallel programs for rank sorting using Bird-Meertens formalism. First, a general unbounded parallel program is constructed. Then, we transform the program, by imposing a limited number of processors. Three variants are obtained, two for the case $p \leq n$ and one for the case $n < p \leq n^2$. Then the implementations of the programs onto shared memory (SM), distributed memory (DM), and pipeline architectures are analyzed. If $p \leq n$ the first BMF variant is used for implementation on SM architectures; for DM and also for pipeline architectures the second variant is most appropriate. A

pipeline architecture with more than n processors for this problem is not useful. If $n < p \leq n^2$ the implementations on SM and DM architectures start from the same BMF program. For the DM case data replication is going to be used.

Time complexities for each case are analyzed too.

Parallel programs for rank sorting have been developed before, rank sorting algorithm being one that shows that a poor sequential algorithm for a problem may lead to very good parallel algorithms. It has been considered especially for sorting on SM machines [7]. We have shown here in a formalized way that we may successful use it for DM and pipeline architectures. Also, we have formally analyzed the case when the number of processes p is between n and n^2.

Yet, with this example, a more important thing has been emphasized: BMF programs can be formally transformed for bounded parallelism and the variants which are obtained may be analyzed later for choosing the right one for a specific target machine.

The abstract design using BMF can comprise different cases that may appear at the implementation phase: shared memory versus distributed memory and pipeline computation, and different numbers of processors. So, the abstractness of this formalism does not exclude performance.

References

[1] R. Bird. *Lectures on Constructive Functional Programming.* In M. Broy editor, Constructive Methods in Computing Science, NATO ASI Series F: Computer and Systems Sciences, Vol. 55, pg. 151-216. Springer-Verlag, 1988.

[2] M. Cole. *Parallel Programming with List Homomorphisms.* Parallel Processing Letters, 5(2):191-204, 1994.

[3] Gorlatch, S., *Abstraction and Performance in the Design of Parallel Programs*, CMPP'98 First International Workshop on Constructive Methods for Parallel Programming, 1998.

[4] Knuth, D.E., *The Art of Computer Programming. Vol. 3 Sorting and Searching*, Addison-Wesley, 1973.

[5] D.B. Skillicorn. *Foundations of Parallel Programming.* Cambridge University Press, 1994.

[6] Skillikorn, D., *Architectures, Costs, and Transformations*, CMPP'98 First International Workshop on Constructive Methods for Parallel Programming, 1998.

[7] Wilkinson, B., Allen, M., *Parallel Programming Techniques and Applications Using Networked Workstations and Parallel Computers*, Prentice Hall, 2002.

Department of Computer Science, Babeş-Bolyai University, Cluj-Napoca
E-mail address: vniculescu@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

ROAMING OPTIMIZATION: A NEW EVOLUTIONARY TECHNIQUE FOR MULTIMODAL OPTIMIZATION

R. I. LUNG AND D. DUMITRESCU

Abstract. A new evolutionary technique for multimodal optimization called *Roa-ming technique (RT)* is proposed. Multiple optima are detected using subpopulations evolving in isolation. A stability measure is defined for subpopulations by which they are characterized as stable or unstable. Stable subpopulations are considered to contain local optima. An external population called *the archive* is used to store the optima detected. After a number of generations the archive contains all local optima. Experimental results prove the efficiency of the algorithm.

1. Introduction

Most real world problems require the detection not only of one global optimum but also of the other global or local optima. In some cases the local optima may be almost as good as the global one, or they may provide the human decision maker with a better insight into the nature of the design space.

Evolutionary algorithms were initially designed to detect global optima and special techniques to avoid premature convergence to local optima have been developped. However, it is only natural due to the intrinsic parallelism of the evolutionary methods to assume that such methods should be able to detect also multiple solutions.

Several evolutionary approaches to the multimodal optimization problem have been made. Among them we mention: fitness sharing [1], crowding [2], deterministic crowding [4], Multinational genetic algorithm [6], Forking genetic algorithm [5] and the adaptive elitist-population based genetic algorithm [3].

2. Roaming optimization

A new evolutionary technique for multimodal optimization called *Roaming technique* is proposed.

Roaming technique identifies the local optima using isolated subpopulations and stores them in an external population called *archive*.

A stability measure for subpopulations is defined. By using it, subpopulations are evaluated as stable or unstable. Unstable subpopulations evolve in isolation until they become stable. The best individual in a stable subpopulation is considered to be a potential local optimum.

The number of subpopulations is a parameter of the algorithm and it is not related to the expected number of local optima. This confers flexibility and robustness to the roaming search mechanism.

Potential optima are saved into the *archive*. Stable subpopulations are spread over the search space in order to detect other optima.

After a certain number of generations the archive contains all optima.

3. Roaming technique

Consider the optimization problem:

$$\begin{cases} \text{maximize } eval(x), \\ x \in S, \end{cases}$$

where S is the solution space and $eval(x)$ is the fitness value of individual x.

Let N be the number of subpopulations. At each generation t the population P is composed of N subpopulations P_i, $i = 1, \ldots, N$.

We may define an order relation on P.

Definition 3.1. We say that individual x *is better then* y, and we write $x \succ y$, if and only if the condition

$$eval(x) \geq eval(y),$$

holds.

3.1. **Stability measure.** A *stability measure* is introduced for determining whether a subpopulation has located a potential optimum.

By evolving subpopulation P_i for n generations a new subpopulation $P_i^{'}$ having the same size as P_i is obtained.

Definition 3.2. The number n of iterations the subpopulations evolve in isolation until their stability is measured is a parameter of the algorithm called the *iteration parameter*.

Let x_i^* be the best individual in the parent subpopulation P_i. We define an operator B as the set of individuals in the offspring in subpopulation $P_i^{'}$ that are better then x_i^*:

$$B : P \longrightarrow \mathcal{P}(P)$$
$$B(x_i^*) = \{x \in P_i^{'} \mid x \succ x_i^*\}.$$

Using the cardinality of the set B a stability measure $SM(P_i)$ of subpopulation P_i may be defined.

Definition 3.3. Stability measure of the subpopulation P_i is the number $SM(P_i)$ defined as

$$SM(P_i) = 1 - \frac{card\, B(x_i^*)}{card\, P_i},$$

where x_i^* is the best individual in P_i and $card\, A$ represents cardinality of the set A.

Proposition 3.4. Stability measure of a subpopulation P has the following properties:

(i) $0 \leq SM(P) \leq 1$;
(ii) If $SM(P) = 1$ then x^* is a potential local optimum;

where x^* is the best individual in P.
Proof. It is obvious using stability measure definition.

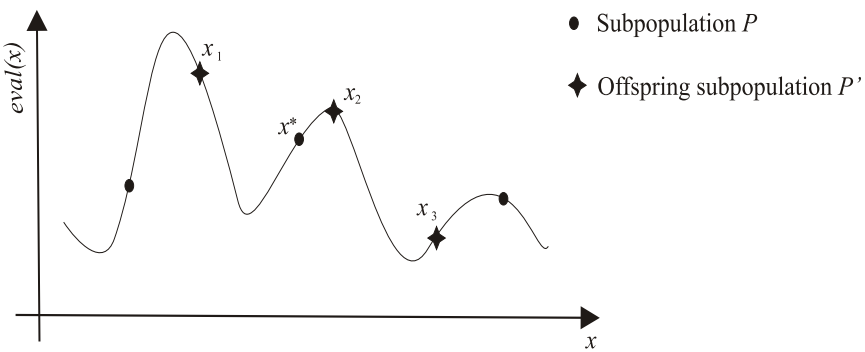

FIGURE 1. Subpopulations P and P' illustrating the stability measure concept

Definition 3.5. A subpopulation P is called *σ-stable* if the condition

$$SM(P) \geq \sigma \tag{1}$$

holds, where $0 \leq \sigma \leq 1$. A 1-stable subpopulation is called a *stable* subpopulation.

Remarks 3.6. The following remarks on subpopulations stability can be made:

(i) A subpopulation P is called σ-ustable if it is not σ-stable.
(ii) The best individual in a stable subpopulation can be considered a potential local optima.

Example 3.7. Suppose subpopulation P contains 3 individuals and x^* is the best of them (Figure 1). The offspring subpopulation P' contains two individuals, x_1 and x_2 that are better then x^*. We have

$$x_1 \succ x^*,\ x_2 \succ x^*,\ \text{but } x^* \succ x_3.$$

Therefore

$$B(x^*) = \{x_1, x_2\}.$$

The stability measure of P is

$$\begin{aligned} SM(P) &= 1 - \frac{card\, B(x^*)}{card\, P} \\ SM(P) &= \frac{1}{3}. \end{aligned}$$

In conclusion, we can say that P is a 1-unstable subpopulation.

3.2. **Archive.** Within Roaming technique 1-unstable subpopulations evolve in isolation until they become stable. The best individual in a stable subpopulation is considered a potential local optimum. An external population called the *archive* is used to store detected potential optima.

Consider a stable subpopulation P, and x^* the best individual in P. Then x^* is considered a potential local optimum.

It is reasonable to suppose that a potential optimum can be a local optimum or can be very close to a local optimum.

Before adding a candidate solution x^* to the archive the distance between x^* and every solution a in the archive is compared with an archive parameter δ. δ is an algorithm parameter and is related to the minimum distance between two optima. If the condition $d(x^*, a) < \delta$ holds then only the best fitted from x^* and a enters the archive.

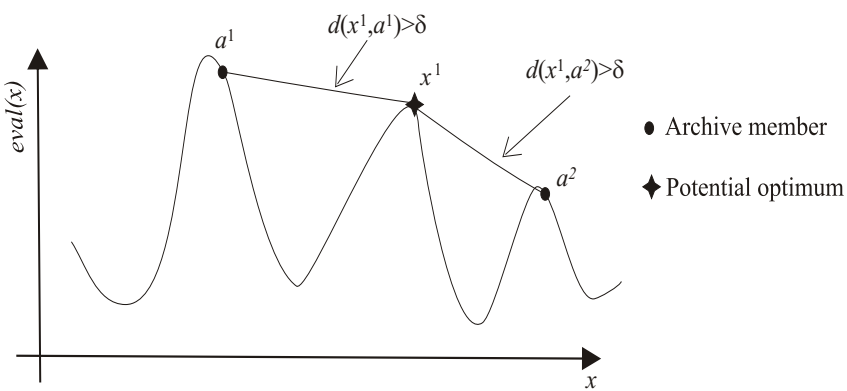

FIGURE 2. Adding a potential optima to the archive case *(i)*: x^1 is included into the archive

Summarizing, a solution x^* is added to the archive if and only if one of the following situations arrize:

(i) the distance to all other solutions in the archive is higher then δ, meaning that x^* represents a new optimum for the archive;

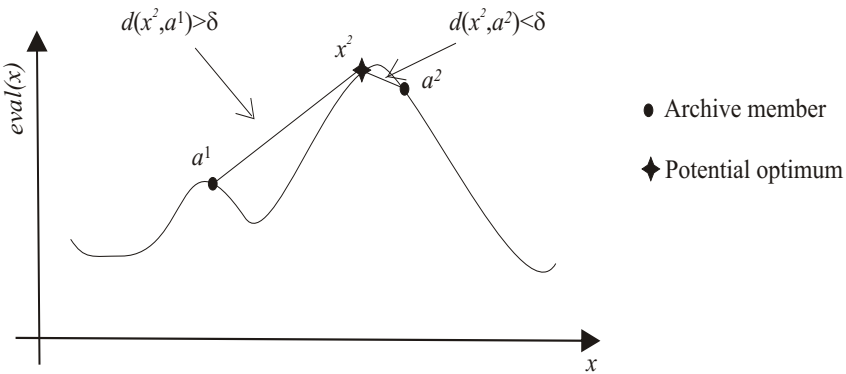

FIGURE 3. Adding a potential optima to the archive case *(ii)*: x^2 is included into the archive

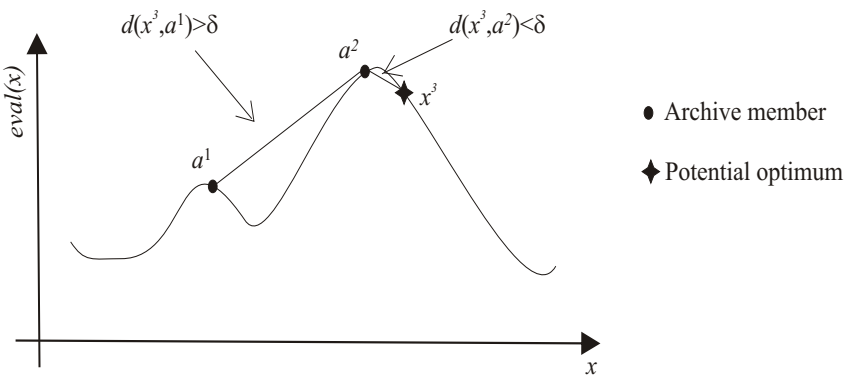

FIGURE 4. Adding a potential optima to the archive case *(ii)*: x^3 is not included into the archive

(ii) x^* is better then a solution a in the archive that is at a distance smaller then δ from x^*, in which case x^* replaces a.

Figures 2 3 and 4 illustrate the two archive addition cases. In the first case (Figure 2) the potential optimum x^1 is added to the archive. This is the case when the distance to all individuals in the archive is higher then δ. In the second case (fig. 3) there is an individual a^2 in the archive within the distance δ. x^2 is

better then a^2, therefore x^2 is added to the archive. Figure 4 describes another situation that can arrize, in which the potential optima x^3 is not added to the archive.

Remark 3.8. The archive is used exclusively to store the potential optima. Solutions preserved in the archive do not participate in further selection processes. Thus the Roaming technique can not be considered as an elitist one.

3.3. **Roaming subpopulations.** Consider a potential optimum x_i^* has been added to the archive. To avoid the search process to get stuck, the search performed by the subpopulation P_i has to be redirected towards other regions of the search space. In this respect stable subpopulations are selected to be spread in the search space in order to detect new optima.

Subpopulations selected for spreading are called *Roaming Subpopulations*. Selection is performed using subpopulations stability measure.

A parameter $RS \in [0, 1]$ is used as a threshold in order to select the stable subpopulations. If for subpopulation P we have $SM(P) > RS$ then it is selected as a roaming subpopulation. Acctually all RS-stable subpopulations are selected. The RS-unstable subpopulations are called *non-roaming subpopulations*.

Roaming subpopulations are spread in the search space in order to detect other optima. The subpopulaiton roaming is realized using variation operators acting on subpopulations. We have used a mutation operator for subpopulations that applies strong mutation to each individual of the subpopulation. To ensure complete change of the individuals of the subpopulations, the mutation rate should be close to 1. Other types of genetic operators for subpopulations can be defined.

3.4. **Computing next generation.** Let $P_i'(t)$ be the offspring subpopulation from $P_i(t)$. The next generation $P(t+1)$ is composed of the roamed subpopulations and the offspring $P_i^{'}(t)$ of all non-roaming subpopulations $P_i(t)$.

3.5. **Roaming algorithm.** Roaming technique evolves several subpopulations in order to detect the local optima of a multimodal problem.

A measure for the stability of a subpopulation is used. Unstable subpopulations evolve in isolation until they become stable. The best individual of a stable subpopulation is considered to be a potential local optima. Potential local optima are saved into an external population called *archive*.

The algorithm stops after a given number of generations. At the end the archive contains detected local optima.

Figure 5 illustrates one iteration of the algorithm for a population $P(t)$ composed of five subpopulations.

Roaming algorithm may be outlined as follows:

Roaming algorithm

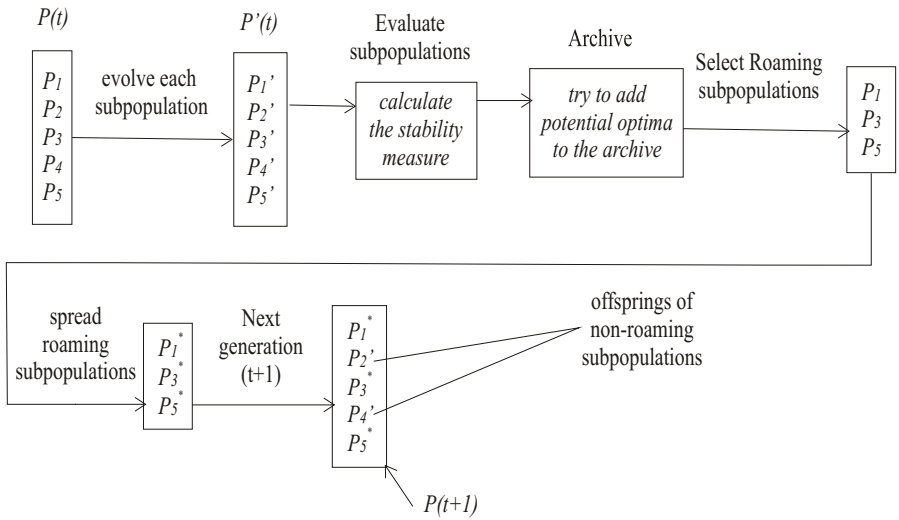

FIGURE 5. An iteration of the algorithm

Input: N - subpopulations number
$Popsize$ - subpopulation size
$Ngen$ - maximum number of generations
n - iteration parameter
δ - archive parameter
RS - roaming threshold
p_c, p_m -crossover probability and mutation rate
Output: $Archive$ - the set of local optima

Step 1. *Initialization:*
a) $t := 0$;
b) Initialize $P_i(t)$ for each $i = 1, \ldots N$ by randomly generating *popsize* number of individuals;
c) $Archive := \emptyset$.
Step 2. Evaluate each individual x in each subpopulation $P_i(t)$ by computing its fitness $eval(x)$;
Step 3. Evolve each subpopulation $P_i(t)$ for n iterations applying selection, recombination and mutation . Let $P_i^{'}(t)$ be the resulting offspring subpopulation.
Step 4. Evaluate each individual x in the offspring subpopulation $P_i^{'}(t)$ by caculating its fitness $eval'(x)$.

Step 5. For each subpopulation $P_i(t)$ calculate:
a) The best individual x_i^*;
b) The stability measure $SM(P_i(t))$ using Definition 3.3.
Step 6. For each subpopulation $P_i(t)$ having $SM(P_i(t))$=0 try to add x_i^* to the *Archive*.
Step 7. For each $i = 1, \dots N$ do if $SM(P_i(t)) \geq RS$ then consider $P_i(t)$ to be a roaming subpopulation;
Step 8. Migrate all roaming subpopulations using strong mutation with $rate = 1$
Step 9. Set $P(t+1) = \{P_i(t) \,|\, P_i(t)$ is a Roaming Subpopulation$\} \cup \{P_i'(t) \,|\, P_i(t)$ is not a Roaming Subpopulation$\}$; $t = t + 1$. If $t < Nrgen$ then go to step 2, else stop.

4. Experimental results

Roaming Algorithm has been tested for several standard functions. In this section the following functions are considered:

- $f_1(x) = e^{-2\ln(2)\left(\frac{x-0.1}{0.8}\right)^2} \sin^6(5\pi x)$, $0 \leq x \leq 1$,
- $f_2(x) = \sum_{j=1}^{5} j \, \cos((j+1)x + j)$, $0 \leq x \leq 10$,
- $f_3(x_1, x_2) = 1 + \sin^2 x_1 + \sin^2 x_2 - 0.1 \cdot e^{(-x_1^2 - x_2^2)}$, $-5 \leq x_1 \leq 5$, $-5 \leq x_2 \leq 5$.

The parameters used to run the algorithm for functions f_1, f_2 and f_3 are presented in Table 1.

Evolution of the archive content for function f_1 the is presented in Figures 6 and 7. Roaming algorithm detects the peaks of the function at early stage of the search process. This can be noticed also for the functions f_2 and f_3 in Figures 8 and 10 where the achive content after 10 generations is presented.

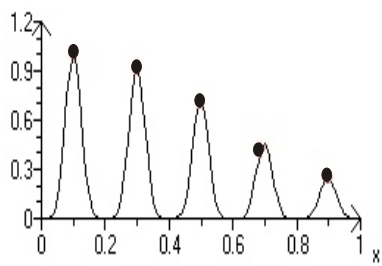

Figure 6. Function f_1 - the archive after 10 generations

Figures 9 and 11 present the final achive for functions f_2 and f_3. The number of optima stored in the archive for each function in presented in Table 2.

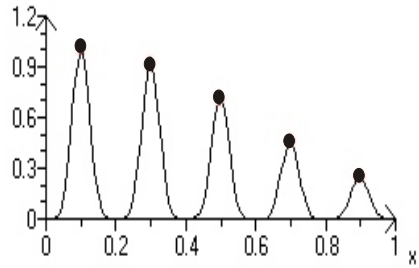

FIGURE 7. Function f_1 - the archive after 75 generations

TABLE 1. Parameters used for f_1, f_2 and f_3

Parameter	f_1	f_2	f_3
Subpopulation number	15	15	10
Subpopulation size	10	10	10
Number of generations	75	75	50
δ	0,1	0,3	2
Iteration parameter	1	1	1
RS	0,8	0,8	0,8
Crossover probability	0,5	0,5	0,5
Mutation rate	0,05	0,05	0,05

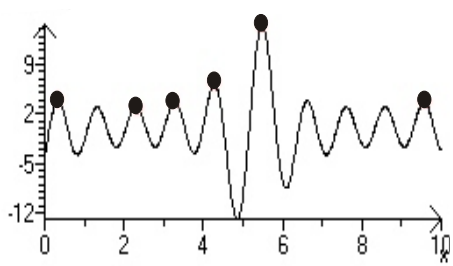

FIGURE 8. Function f_2 - the archive after 10 generations

4.1. **Parameters Setting.** Roaming technique uses specific parameters such as the RS threshold, δ and the number of subpopulations.

The RS parameter is chosen so that only stable subpopulations are selected to roam. Acctually all RS-stable subpopulations are selected to roam.

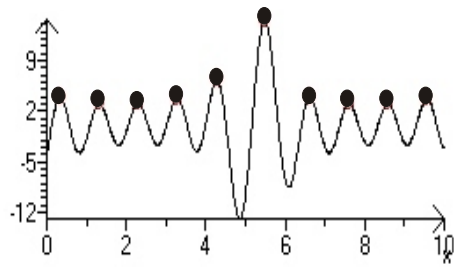

FIGURE 9. Function f_2 - the archive after 75 generations

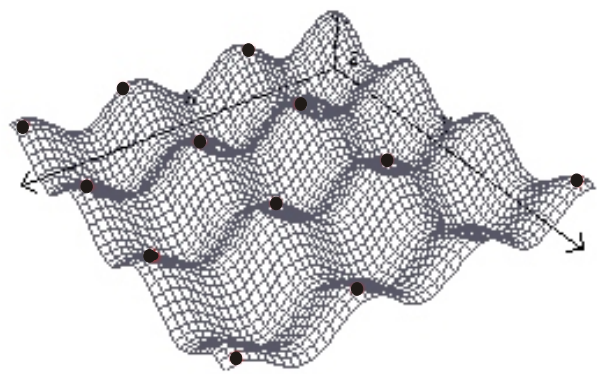

FIGURE 10. Function f_3 - the archive after 10 generations

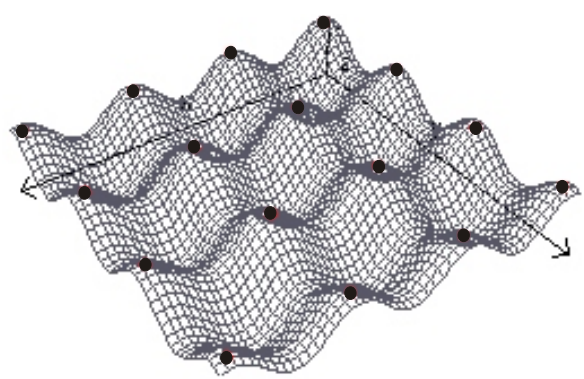

FIGURE 11. Function f_3 - the archive after 50 generations

TABLE 2. Number of peaks detected for functions f_1, f_2 and f_3

Function	Number of generations	Number of detected peaks
f_1	75	5
f_2	75	10
f_3	50	16

The value of δ depends on the distribution of the local optima in the fitness landscape. For evenly distributed landscapes, the choice of δ does not represent a problem. However it is almost impossible to choose a suitable value for δ in the case of the functions with unevenly spaced optima. A mechanism to avoid the use of this parameter is the object of current work.

The algorithm works for any number of subpopulations. If a small number of subpopulations it is used then in order to detect more optima the number of generations has to be increased. Convergence speed depends on the subpopulation number.

5. Conclusions and future work

A new evolutionary technique for multimodal optimization called Roaming is proposed. The technique uses a number of roaming subpopulations in order to detect multiple optima. A measure for the stability of a sub-population is introduced in order to asses whether a subpopulation has located an optimum or not.

Subpopulations evolve in isolation until an optimum is detected. Detected optima are saved into an archive and the corresponding subpopulations are spread towards new promissing regions of the search space.

Numerical examples are presented to illustrate the efficiency of the technique proposed.

References

1. D.E. Goldberg and J. Richardson, *Genetic algorithms with sharing for multimodal function optimization*, Genetic algorithms and their Applications ICCGA'87 (Grefenstette, ed.), 1987, pp. 41–49.
2. K. A. D. Jong, *An analysis of the behaviour of a class of genetic adaptive systems*, Ph.D. thesis, University of Michigan, 1975.
3. Kwong-Sak Leung and Yong Liang, *Adaptive elitist-population based genetic algorithm for multimodal function optimization*, GECCO 2003, LNCS 2723 (E. Cantu-Paz et al., eds.), 2003, pp. 1160–1171.
4. S. W. Mahfoud, *Niching methods for genetic algorithms*, Ph.D. thesis, University of Illinois at Urbana Champaign, Illinois Genetic Algorithm Laboratory, 1995.
5. S. Tsutsui, Y. Fujimoto, and A. Gosh, *Forking genetic algorithms: GAs with search space division*, Evolutionary computation **5** (1997), 61–80.
6. R. K. Ursem, *Multinational evolutionary algorithms*, Congres of Evolutionary Computation, 1999, pp. 1633–1640.

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

JAMES A. STORER, "AN INTRODUCTION TO DATA STRUCTURES AND ALGORITHMS", BIRKHÄUSER BOSTON, C/O SPRINGER-VERLAG, NEW YORK, 2002

CLARA IONESCU

This book deals with one of the most challenging subjects of Computer Science. A lot of universities from the whole world are using such books during preparing their students in Computer Sciences. One of the most famous of these books is Introduction to Algorithms written by T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein from MIT. So, what is new in this book, written by James A. Storer, from the Department of Computer Science of the Brandeis University, USA?

First of all it is a wonderful support for students who are attending accompanying lectures on data structures and algorithms. Except for the introduction, exercises, and notes for each chapter, page breaks have been put in manually and the format corresponds to a lecture style with ideas presented in "digestible" page-size quantities.

Algorithms are presented with pseudo-code, not programs from a specific language, let the reader to implement them in which language he or she whish to do it.

Whole the book can be cut in some parts in order to "construct" courses for different kind of interests. A first course on data structures and algorithms may be based on Chapters 1 through 4 (RAM model, Lists, Induction and Recursion and Trees), along with portions of Chapter 5 (Algorithm Design), the first half of Chapter 6 and 12 (Hashing and Graphs) and with parts of Chapter 11 (Strings). Chapters 1 through 4 cover more elementary material, and at a slower pace, the concise style of this book makes it important, that the teachers provide motivation, discuss exercises, and assign homework. For upper class undergraduates the course can essentially start with the Chapter 5, where are presented algorithm design techniques divide and conquer, dynamic programming, randomization, greedy algorithms, graph algorithms etc. This study can be continued with contents of Chapter 7 (Heaps), of Chapter 8 (Balanced Trees), 9 (Sets over a Small Universe) and Chapter 12 (Discrete Fourier Transform).

There is no chapter on sorting. Instead, sorting is used in many examples, which include bubble sort, merge sort, tree sort, heap sort, quick sort, and several parallel

Received by the editors: June 15, 2004.

sorting algorithms. There is no chapter on NP-completeness. Formal treatment of complexity issues like this is left to a course on the theory of computation.

The last chapter presents the PRAM model for parallel computation. The basic concepts presented in the chapter can provide a foundation for further study on specific practical architectures.

The book contains a rich Appendix of Common Sums, a bibliography with more than 600 entries and Index. Unfortunately the bibliography's entries are not numbered and are not referred in the text.

Department of Computer Science, Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Cluj-Napoca, Romania

E-mail address: clara@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

APOLOGY ON PLAGIARISM PAPERS

THE EDITORS

Since the preceding issue has been send to print we have found out, and have been informed by more interested readers that the following papers are plagiates:

- D. Marcu, *The Chromatic Number of Triangle-Free Regular Graphs*, Studia Universitatis Babeş-Bolyai Series Informatica, 47 (1), 2002, p. 54–56.
- D. Marcu, *A Note on the Chromatic Number of a Graph*, Studia Universitatis Babeş-Bolyai Series Informatica, 47 (2), 2002, p. 105–106.
- D. Marcu, *A Note on the Chromatic and Independence Number of a Graph*, Studia Universitatis Babeş-Bolyai Series Informatica, 48 (2), 2003, p. 11–16.

According to practices currently in place, these papers have been reviewed, as always, by a panel of two experts. They have made all possible effort to ensure the scientific quality and accuracy of the papers submitted to the journal. However, we are not always able to verify the originality of every paper submitted, and, as usually, this rests with the responsibility of the author.

After a careful consideration, we have decided to retract the papers under scrutiny; the papers will be marked as such on the journal web page. As we have lost the confidence in Mr. Dănuţ Marcu, the author of these plagiates, we have decided to ban Mr. Marcu from publishing in our journal.

We are apologizing to the international scientific community for this situation. Despite this, we are ensuring our readers that we are continually working to ensure a high scientific quality for our journal. As such, they may continue to consider our journal as the journal of their choice.

The Editors

Faculty of Mathematics and Computer Science, Department of Computer Science, Babes-Bolyai University, 400084 Cluj-Napoca, Romania

E-mail address: studia-i@cs.ubbcluj.ro

Received by the editors: May 15, 2004.

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLIX**, Number 1, 2004

APOLOGY ON PLAGIARISM PAPERS

THE EDITORS

Since the preceding issue has been send to print we have found out, and have been informed by more interested readers that the following papers are plagiates:

- D. Marcu, *The Chromatic Number of Triangle-Free Regular Graphs*, Studia Universitatis Babeş-Bolyai Series Informatica, 47 (1), 2002, p. 54–56.
- D. Marcu, *A Note on the Chromatic Number of a Graph*, Studia Universitatis Babeş-Bolyai Series Informatica, 47 (2), 2002, p. 105–106.
- D. Marcu, *A Note on the Chromatic and Independence Number of a Graph*, Studia Universitatis Babeş-Bolyai Series Informatica, 48 (2), 2003, p. 11–16.

According to practices currently in place, these papers have been reviewed, as always, by a panel of two experts. They have made all possible effort to ensure the scientific quality and accuracy of the papers submitted to the journal. However, we are not always able to verify the originality of every paper submitted, and, as usually, this rests with the responsibility of the author.

After a careful consideration, we have decided to retract the papers under scrutiny; the papers will be marked as such on the journal web page. As we have lost the confidence in Mr. Dănuţ Marcu, the author of these plagiates, we have decided to ban Mr. Marcu from publishing in our journal.

We are apologizing to the international scientific community for this situation. Despite this, we are ensuring our readers that we are continually working to ensure a high scientific quality for our journal. As such, they may continue to consider our journal as the journal of their choice.

The Editors

Faculty of Mathematics and Computer Science, Department of Computer Science, Babes-Bolyai University, 400084 Cluj-Napoca, Romania

E-mail address: studia-i@cs.ubbcluj.ro

Received by the editors: May 15, 2004.