REVISTA INVESTIGACIÓN OPERACIONAL VOL. 37, NO. 2, 173-183, 2016

MAN POWER OPTIMIZATION IN LARGE-SCALE CORPORATIONS

Pablo Romero, Franco Robledo, Patricia Bevilacqua and Martín Delafuente
Facultad de Ingeniería, Universidad de la República*

ABSTRACT
Manpower optimization and job scheduling represent key elements in the Work Force Management (WFM) field. Large-scale corporations face a trade-off, where the staff is bounded to achieve savings but customers should receive the service effectively and timely.
A combinatorial optimization problem for contract and job scheduling is introduced, inspired in large-scale corporations. The main simplifications in this *Contract-Scheduling* problem are multi-class Poissonian customer arrivals, exponential services, one-to-one attention and fixed salaries from the staff. The goal is to minimize the staff respecting multiple timing constraints (coming from customers), and define atomic units inside the corporation that offer independent services.
Even under these simplifications, the problem belongs to the class of $\mathcal{NP}$-Hard decision problems. Specifically, the feasibility of Contract-Scheduling problem includes the determination of the chromatic number of a graph.
As a consequence, a heuristic methodology is developed for its resolution, combining Monte Carlo simulation and a greedy-randomized technique. Numerical results confirm an intuitive contract approach is true: units should be stressed offering the maximum number of affine services, and the staff should be kept as small as possible, respecting demand constraints.

KEYWORDS: Manpower Optimization, Computational Complexity, Monte Carlo Simulation

MSC: 90C27; 90C59.

RESUMEN
La optimización de la fuerza laboral y la planificación del trabajo representan ele- mentos estratégicos en la gestión empresarial. Las empresas de alto porte afrontan el compromiso de rentabilidad, donde se debe mantener una cantidad de personal limitada y al mismo tiempo los clientes deben recibir sus servicios efectivamente y a tiempo. En este artículo se presenta un problema de optimización combina- toria para el contrato y la asignación del trabajo, inspirado en empresas de alto porte. La principal simplificación en este problema de Contrato-Asignación con- siste en arribos Poisson de múltiples clases de clientes, servicios que responden a leyes exponenciales independientes, con atención uno a uno y salarios fijos del personal. El objetivo es minimizar la cantidad total de personal respetando restric- ciones temporales (provenientes de la tolerancia de los clientes), y definir unidades atómicas dentro de la empresa que ofrecen servicios independientes. Incluso bajo estas simplifcaciones, el problema pertenece a la clase NP-Difícil. Específicamente, la factibilidad del problema de Contrato-Asignación incluye la determinación del número cromático de un grafo. Como consecuencia, se desarrolla una heurística para su resolución, que combina simulación por Monte Carlo y una técnica golosa aleatorizada. La intuición sugiere que es preferible estresar unidades, ofreciendo la máxima cantidad de servicios posible en cada una, respetando al mismo tiempo restricciones temporales. Los resultados experimentales sugieren que esta intuición es correcta.

*(promero,frobledo)@fing.edu.uy; (martin.delafuente,patricia.bevilacqua)@gmail.com

1. INTRODUCTION

Manpower optimization is a field of knowledge that supports operational decisions in the leader or manager of a corporation. A key element is to clearly determine the service offered by the corporation, its correlation, the units or independent atoms that jointly compose the corporation and the whole staff or contract. These decisions, together with innovative ideas, personnel skills and world economics in terms of investment, determine the success of a current modern corporation. In literature reviews we can find general qualitative tips, spider-graphs for the design of a corporation, and others [13, 5]. However, little effort has been focused on a mathematical framework to develop a full corporation [12, 2], and most works are focused on special cases, as call centers [3].

The goal of this paper is to present a novel and simple mathematical framework to understand the dimensioning and design of a corporation at its operational level. By means of a combinatorial optimization problem, we formulate the minimum cost design of a corporation, subject to Poissonian multi-customer arrivals, independent exponential time services and minimum quality of service (reduced waiting times) to customers. This model also considers the level of correlation of different services, and role assignment for different units of the corporation to be deployed.
This paper is structured as follows. Section 2. presents the terminology of graphs and complexity theory that will be used throughout the treatment. Section 3. presents a mathematical formulation of the Contract-Scheduling Problem (CSP), while Section 4. states that the CSP belongs to the class $\mathcal{NP}$-Hard problems. As a consequence, we develop a heuristic resolution, combining Monte Carlo simulation and a greedy notion for the CSP, in Section 5..
In Section 6., we illustrate a coherent property of the model with sample scenarios: it is better to stress the system, maximizing the probability of occupation for the staff. Additionally, we highlight the fact that the number of atoms (i.e. independent units) of the corporation should be minimized. This result is intuitive, and the theoretical foundation is highly connected with queuing systems. Concluding remarks and trends for future work are covered in Section 7..

2. TERMINOLOGY

In this section, the terminology used throughout this article in graph theory and computational complexity is provided.

2.1. Computational Complexity

The class $\mathcal{NP}$ is the set of problems polynomially solvable by a non-deterministic Turing machine [6]. A problem is $\mathcal{NP}$-Hard if it is at least as hard as every problem in the set $\mathcal{NP}$ (formally, if every problem in $\mathcal{NP}$ has a polynomial reduction to the former). It is widely believed that $\mathcal{NP}$-Hard problems are intractable (i.e. there is no polynomial-time algorithm to solve them). An $\mathcal{NP}$-Hard problem is $\mathcal{NP}$-Complete if it is inside the class $\mathcal{NP}$. Stephen Cook proved that the joint satisfiability of an input set of clauses in disjunctive form is an $\mathcal{NP}$-Complete decision problem; in fact, the first known problem of this class [4]. In this way, he provided a systematic procedure to prove that a certain problem is $\mathcal{NP}$-Complete. Specifically, it suffices to prove that the problem is inside the class $\mathcal{NP}$, and that it is at least as hard as an $\mathcal{NP}$-Complete problem. Richard Karp followed this hint, and presented the first 21 combinatorial problems inside this class [8]. In general, given a class $\mathcal{C}$ of problems, the problem P is $\mathcal{C}$-complete if $P \in \mathcal{C}$ and it is at least as hard as all problems in the class $\mathcal{C}$.

A combinatorial optimization problem (COP) is to $\min_{x \in S} f(x)$, where f is the objective function and $x \in S$, a non-empty finite set of solutions ($\arg\max_{x\in S} -f(x) = \arg\min_{x \in S} f(x)$, maximization problems are also COPs). The domain for x is some finite set $D : S \subseteq D$, and $x \in D$ is called a solution (it is feasible provided $x \in S$).
Every COP has a corresponding decision problem:

Definition 2.1. $(COP - r - QUALITY)$. *Given a COP* $\max_{x \in S} f(x)$ *and a real number* r. *Does there exist a feasible solution* x *such that* $f(x) \leq r$*?*

2.2. Graph Theory

A *simple graph* is a pair $G = (V, E)$, where V is a nonempty set and E a symmetric binary relation on V. If $(x, y) \in E$, we say that x and y are *adjacent* vertices. Let us denote $[k] = \{1, \ldots, k\}$, for any positive integer k. A *k-coloring* of G is a function $f : V \to [k]$ such that $f(v) \neq f(w)$ whenever $\{v, w\} \in E$. Its *chromatic number* $\kappa(G)$ is the least positive integer such that there exists a κ-coloring (i.e. there exists a graph-coloring $f : V \to [\kappa]$). A subset $V' \subseteq V$ is a *vertex-cover* for G if all edges from the set E are adjacent to some $v \in V'$. A subset of vertices $V^* \subseteq V$ is *independent* if there is no pair of vertices $v_1, v_2 \in V^*$ such that $(v_1, v_2) \in E$.

The two following problems are included in Karp list of 21 combinatorial problems [8]. They have a strict relation with Contract-Scheduling problem.

Definition 2.2. $(K - COLORABILITY)$. *Given a simple graph* $G = (V, E)$ *and a positive integer* K. *Does there exist an assignment* $f : V \to \{1, \ldots, K\}$*?*

Definition 2.3. $(VERTEX - COVER)$. *Given a simple graph* $G = (V, E)$ *and a positive integer* K. *Does there exist a vertex-cover* $V' \subseteq V$ *with* $|V| \leq K$*?*

3. CONTRACT-SCHEDULING PROBLEM

Consider a large corporation that offers services $s = (s_1, \ldots, s_n)$. Customers are grouped into n disjoint classes (one per service), and their arrivals follow a Poissonian process with respective intensity rates $\lambda = (\lambda_1, \ldots, \lambda_n)$. They expect to be attended in not more than a_i time units (seconds, minutes or even days, according to the service). On the other hand, the staff offers the service following exponentially distributed times, with respective rates $\mu = (\mu_1, \ldots, \mu_n)$.
The staff cannot cope with all requests (services) at the same time, mainly because these services can be weakly correlated. As a consequence, a binary matrix $M \in \{0, 1\}^{n \times n}$ assumes $m_{ij} = 1$ whenever services s_i and s_j can be offered by the same person, and $m_{ij} = 0$ otherwise. Services are assigned to (and internally managed by) K atomic units of the corporation, with respective number of staff $p_1, \ldots, p_K$. Denote $S \in \{0, 1\}^{K \times n}$ the *scheduling matrix*, such that $s_{ij} = 1$ whenever atom i is involved in service j, and $s_{ij} = 0$ otherwise. All services must be attended by exactly one atom.
Customers are attended following a first in-first out (FIFO) queuing policy in a face-to-face communication with some member of the staff. Specifically, the customer selects the correct atom, and looks for an idle worker. If there is no worker available, he/she should wait in a FIFO queue. On the other hand, workers are available when requested and not busy, and they are able to offer all services assigned to the atom where they work.

Let $p = (p_1, p_2, \ldots, p_K)$ the *contract vector*, or complete staff, and K the *size*, or number of atoms inside the corporation. If we further assume identical salaries, it is clear that the operator should minimize the staff in order to reduce the operational costs. The Contract-Scheduling Problem (CSP) is to decide the corporation size K, contract vector p and scheduling matrix S, respecting both timing constraints from customers and affine services:

$$\min_{K,p,S} p = \sum_{i=1}^{K} p_i \tag{1}$$

$$s.t.$$

$$\sum_{i=1}^{K} s_{ij} = 1, \, \forall j \in \{1, \ldots, n\} \tag{2}$$

$$s_{ij}s_{ik} \leq m_{jk}, \, \forall i \in \{1, \ldots, K\}, j, k \in \{1, \ldots, n\} \tag{3}$$

$$E(T_i) \leq a_i, \, \forall i \in \{1, \ldots, n\} \tag{4}$$

The objective (1) is to minimize the staff of the corporation. Constraint (2) states that every service must be offered exactly by one atom. Constraint (3) confirms that the services offered by a certain atom must be affine. Finally, Constraint (4) deserves further explanation, and states that the expected sojourn time $E(T_i)$ experienced by a customer from class-i must not exceed the threshold a_i.
When an atom with n customers offers a single service s_i we have a single-class waiting system with FIFO queuing policy, and there is an explicit expression for $E(T_i)$ based on Erlang C Formula. If $A = \frac{\lambda}{\mu}$ denotes the offered traffic then the probability of a customer to have a positive waiting time equals $E_{2,n}$:

$$E_{2,n} = \frac{\frac{A^n}{n!}\frac{n}{n-A}}{\sum_{i=0}^{n-1}\frac{A^i}{i!} + \frac{A^n}{n!}\frac{n}{n-A}}. \tag{5}$$

After algebraic manipulation of cut-equations under stationary state [1], the mean queuing length L_n can be found:

$$L_n = E_{2,n}\frac{A}{n-A} \tag{6}$$

Finally, using Little's law [9] it turns out that the mean sojourn time T_n is related with Erlang's C Formula:

$$E(T_n) = \frac{L_n}{\lambda} + \frac{1}{\mu} = E_{2,n}\frac{1}{n\mu - \lambda} + \frac{1}{\mu}. \tag{7}$$

Expression (7) is precisely the left-hand of Constraint (4) with a singleton service associated to a certain unit in the corporation.

To the best of our knowledge, there is no closed-form for $E(T_i)$ in a multi-class poissonian arrival system with exponentially distributed services, following a non-preemptive FIFO waiting policy. For this reason, atoms offering multiple services are studied numerically here, using Monte Carlo simulation to find an unbiased estimator for $E(T_i)$.

4. PROBLEM COMPLEXITY

We will elucidate a connection between the scheduling part of the CSP (Constraints (2) and (3)) and a classical decision problem coming from graph theory, called $K - COLORABILITY$.

First, consider $CSP1$, which is Contract-Scheduling problem when each atom is assigned exactly one service. It has several differences with CSP. The first one is that the staff is the only decision variable. Other relevant difference with CSP is that an explicit formulation is available for $CSP1$:

$$\min_{p_i} p = \sum_{i=1}^{n} p_i \tag{8}$$

$$s.t.$$

$$E_{2,p_i} \frac{1}{p_i\mu - \lambda} + \frac{1}{\mu} \leq a_i, \, \forall i \in \{1, \ldots, n\} \tag{9}$$

Last but not least, a direct resolution is feasible for the $CSP1$. Just add staff p_i in each atom until Constraint (9) is respected.

However, the resolution of $CSP1$ gives no hint to find the optimum for the CSP. Indeed, the following result holds from elementary queuing theory:

Proposition 4.1.. *The sojourn time of a customer under an $M/M/2$ system with arrival intensity λ is lower than the same system under two separated $M/M/1$ lines, each one with intensity rate $\lambda/2$.*

We can read Proposition 4.1. in terms of the CSP as follows. If we assume compatibility between services s_i and s_j, and further the rates respect $\lambda_i = \lambda_j$ and $\mu_i = \mu_j$, then if we link services i and j together, we will find a better solution for the CSP than singleton services in different atoms.
Therefore, we have an intuition that it is better to "group" the most number of services in the same unit. If we identify "colors" to different units of the corporation, two services can have the same color if and only if they are compatible. This elucidates a connection between $K-COLORABILITY$ and $CSP-K-QUALITY$.

Theorem 4.1.. *$CSP-r-QUALITY$ belongs to the class of $\mathcal{NP}$-Hard decision problem.*

Proof. We will show that $r-COLORABILITY$ is included in $CSP-r-QUALITY$. Consider a corporation with vanishingly small arrivals and extremely large service rates. Therefore, Constraint (4) is fulfilled with only one worker, and the sum-staff equals the corporation size K in the constructed instance. Non-compatible services (nodes) should be assigned to different units (colors). Finding the least number of units (colors) is precisely the chromatic number of a graph. As a consequence, $r-COLORABILITY$ is included in $CSP-r-QUALITY$, and $CSP-r-QUALITY$ is an $\mathcal{NP}$-Hard decision problem. $\square$

5. HEURISTIC RESOLUTION

In order to understand the structure of the problem, a greedy resolution is first defined, and then we tune it with randomization and sophistication to get higher performance. Theorem 4.1. gives a one-to-one correspondence between graph-coloring and scheduling. Consider the graph $G = (V, E)$ of *unrelated services*, where its nodes are the services $V = \{s_1, \ldots, s_n\}$, and nodes s_i and s_j share and edge if and only if $M_{ij} = 0$ (this is exactly when these services cannot be offered by the same unit). By Theorem 4.1., every graph-coloring for G respect Constraints (2) and (3). Reciprocally, all service assignment into units are represented by a graph-coloring for G, where services are nodes and units are different colors for each node. As a consequence, the *scheduling block* consists of Constraints (2) and (3), whereas the *contract block* consists of Constraints (4).

A greedy notion for this problem is to attack first the Scheduling block (find a graph coloring for G), and then to assign the lowest feasible number of staffs in each unit respecting the contract block. Observe that customers are organized by a queuing system. A classical result from queuing theory is that the sojourn time in a single $M/M/1$ queue with arrival rate $n\lambda$ is better than the sojourn time under n independent $M/M/1$ queues with intensity rates λ. In the CSP, the number of queues is exactly the size of the corporation (i.e. the number of units). From Theorem 4.1., this number is minimized when we find the chromatic number for the graph with unrelated services G, so we should find a graph coloring for G using the minimum number of colors.
Algorithm *SolveCSP* returns a feasible solution for the CSP. It receives the vector rates λ, μ, patience vector a, compatibility matrix M, and returns the pair (S, p) (note that the corporation size K is the number of rows from S). Line 1 constructs the graph G of unrelated services given the compatibility matrix M. Function *Complement* simply describes the graph G with adjacency matrix $\overline{M}$, this is, the complementary binary matrix of M. In Line 2, a feasible graph-coloring for G is found. Even though there are elementary ways to find a graph coloring, the heart and sophistication of this algorithm is hidden in Function *GraphColoring*. Different implementations for *GraphColoring* will be cited in following paragraphs. Once a feasible scheduling is found, we know the services assigned for each atom of the corporation. In Lines 4 to 6 (i.e. for cycle) the simulation is carried-out, in order to find the minimum staff for each atom. Each unit-staff p_i is fixed as the minimum possible (starting from $p_i = 1$ in Line 3), and tuned in Lines 4 and 5 to the minimum that fullfils Constraint. Function $MonteCarlo$ contains an explicit end-to-end simulation of the whole corporation. Given a staff p_i for atom i, it checks whether $E(T_i) \leq a_i$ or not, for a fixed staff p_i. The lowest staff p_i that respects the family of Constraints (4). Each unit staff is tuned by a classical bipartition technique, starting from $p_i = 1$ and doubling in each iteration, until Condition (4) is met for some power k_i, i.e. $p_i = 2^{k_i}$. The root is finally found between 2^{k_i-1} and 2^{k_i} by bipartition.

Algorithm 1 $(S, p) = SolveCSP(a, \lambda, \mu, M)$

```
1: $G = Complement(M)$
2: $S = GraphColoring(G)$
3: $p \leftarrow 1_K$
4: for $i = 1$ TO $K$ do
5:     $p_i \leftarrow Bipartition(MonteCarlo(p_i, a_i, \lambda_i, \mu_i))$
6: end for
7: return $(S, p)$
```

Now, let us focus on possible implementations for *GraphColoring*. There are two wide approaches for graph coloring: finding either sequential or independent sets (or mixtures). A sequential graph-coloring considers a certain ordering on their vertices and assigns colors in that order to the vertices, trying to use the minimum number of colors. The other approach tries to find packages of independent node-sets, and assigns a different color for each package.
Theoretical properties of the chromatic number and graph coloring are detailed in Handbook of Combinatorics [7]. The reader interested in algorithms in graph coloring is invited to consult a recent survey of graph coloring in [11].
In this section we will sketch two independent-based algorithms for graph coloring, called Lazy Recursive Largest First [10], or $LRLF$, and $EXTRACOL$.

$LRLF$ is a greedy algorithm based on the classic Recursive Largest First, or RLF for short [10]. Basically, it implements the same coloring strategy that the classic one, but exploits possible savings computations during runtime and performs faster than the original algorithm due to Frank Thomson Leighton [14]. The *RLF* pseudocode is sketched as follows:

Algorithm 2 $C = RLF(G)$

```
1: while $|V| > 0$ do
2:     $P \leftarrow V$
3:     $U \leftarrow$
4:     $AddNewColor(C)$
5:     $v_c \leftarrow MaxInducedDegree(G, P, V)$
6: end while
7: while $|P| > 0$ do
8:     $Coloring(C, v_c)$
9:     $MoveAdjacents(v_c, P, U)$
10:    $ReduceGraph(G, v_c)$
11:    $v_c \leftarrow MaxInducedDegree(G, P, U)$
12: end while
13: return $C$
```

The general idea of this algorithm is coloring as many vertex as possible with the same color per iteration, building packages of independent sets in each iteration. It begins with choosing the vertex with maximum induced degree from P taking account another set (V if U is empty or U in other case). All vertex inside do/while loop are assigned the same color (line 7). Once this is done, the algorithm moves the adjacent vertex from P to U (line 8) and removes the processed one from G, including its edges (node induction). Unlike *RLF*, *LRLF* optimizes the way of getting the best vertex for each step calculating the induced degree in a lazy manner inside Function *MaxInducedDegree*. For further details in *RLF* and *LRLF*, the reader is referred to [10].

On the other hand, *EXTRACOL* algorithm is designed to coloring large graphs [15]. In fact, the authors recommend its application to graphs with one thousand nodes or more. It is based on "reduce and solve" approach, and combines an initial preprocessing stage that shrinks the graph until its small enough to apply efficiently a coloring algorithm to the residual graph. Roughly, the algorithm implements the following pseudocode:

Algorithm 3 $C = Extracol(G)$

```
1: while $|V| > q$ do
2:     $D \leftarrow DisjointIndependetSets(G)$
3:     $ReduceGraph(G, D)$
4:     $AddNewColor(C_1, D)$
5: end while
6: $C_2 \leftarrow ColorResidualGraph(G)$
7: $C \leftarrow MergeColoring(C_1, C_2)$
8: return $C$
```

Typically the algorithms based on reduction notion extracts one independent set per iteration. The block of Lines 1 to 5 (while loop) implements a new way to build independent sets. Instead of getting an independent

set at a time, Function $DisjointIndependetSets$ returns a set containing the greatest number of pairwise disjoint independent sets (Line 2), in order to maximize the vertex removing process (line 3) and getting the smallest residual graph for the next iteration step. Finally, all vertex involved in independent sets contained in D can be assigned the same color due to its disjoint nature (Line 4).
Function $DisjointIndependetSets$ does the hard work by combining a Taboo Search algorithm called ATS (Adaptative Taboo Search) to get de independent sets, with Set Theory properties to determine if such sets are disjoint. The independent set extraction and graph reduction strategies can be seen in [15]. On the other hand, the second block of Lines 6 and 7 takes the residual graph from the previous block and applies any coloring procedure (Line 6). The performance of Function $ColorResidualGraph$ determines the quality of the solution. In fact, the author designed a competitive memetic algorithm called $MACOL$ [15].

6. RESULTS

We will illustrate the effectiveness of the heuristic here introduced, taking $GraphColoring$ either as $EXTRACOL$ or $LRLF$. The main purpose is to confirm or reject our intuition that it is better to define corporations with the least number of units (which is precisely the chromatic number in graphs), or not.
In order to carry-out different executions, an 8-core, 8GB RAM PC was used, with OMNeT++ Network Simulator Framework (to develop Monte Carlo simulation), C++ as programming language.
Additionally, in order to respect the nature of a massive corporation, we took DIMACS instances of graphs with at least one thousand nodes. We test an heterogeneous variety of instances including random graphs (with prefix DSCJ), flat graphs (prefix "flat") and large random graphs (prefix C), each of them representing compatibility matrices.
Table 1 summarizes the basic information of the fourteen different tests, where we introduce all graphs to $LRLF$ and $EXTRACOL$. The large running times represent the whole time of customers arrivals and services in Monte Carlo simulation, using OMNet++. It is worth to mention that the real CPU execution time lasts a few hours. The key is that the simulation is fastened, and the same conclusions hold as in real time executions. In order to have a faithful comparison, the virtual simulation times and customers inter-arrival times are identical for both $EXTRACOL$ and $LRLF$.
Table 2 presents the results obtained for the corresponding instances. It can be appreciated that $EXTRACOL$ outperforms $LRLF$ in almost all cases in the sense that the number of colors used in EXTRACOL is lower than using LRLF except for the DSCJ100.9 instance, where LRLF got lesser amount of colors (one less). Additionally, the whole staff in the corporation is consistently lower under $EXTRACOL$ even for the instance mentioned above, that is why we think that this small difference may be due to the random nature of certain aspects of the algorithm and for differences in how both coloring strategies perform grouping tasks in units. This suggests that our intuition is true: it is better to define corporations with the least number of units. Further experiments confirm this intuition.

7. CONCLUDING REMARKS

In this paper a combinatorial optimization problem, called Contract-Scheduling Problem, was introduced. It joints two stages of Scheduling (i.e. assignment of services to unit-atoms of the corporation) and Contract (i.e. to determine the required staff for each atom), offering a minimum prestablished level of service to customers. A special case of Contract-Scheduling occurs when the assignment stage is one-to-one, this is, each atom is assigned exactly one service ($CSP1$). Inspired in the complexity of the determination of

Table 1: Fourteen executions, covering seven DIMACS instances, *EXTRACOL* and *LRLF*.

Instance	*Coloring*	Sim. Time (h)	Inter-Arrivals (h)
DSJC1000.1	*LRLF*	24302	$0,25$
DSJC1000.1	*EXTRACOL*	24302	$0,25$
DSJC1000.5	*LRLF*	6232	$0,39$
DSJC1000.5	*EXTRACOL*	6232	$0,39$
DSJC1000.9	*LRLF*	6232	$0,39$
DSJC1000.9	*EXTRACOL*	6232	$0,39$
flat1000_50_0	*LRLF*	6232	$0,39$
flat1000_50_0	*EXTRACOL*	6232	$0,39$
flat1000_60_0	*LRLF*	6232	$0,39$
flat1000_60_0	*EXTRACOL*	6232	$0,39$
C2000.5	*LRLF*	59897	$0,62$
C2000.5	*EXTRACOL*	59897	$0,62$
C4000.5	*LRLF*	1465	$0,09$
C4000.5	*EXTRACOL*	1564	$0,09$

Table 2: Four executions, covering two DIMACS instances, *EXTRACOL* and *LRLF*.

Colors	Staff	Coloring Time (s)	Execution Time (s)
279	2456	$0,131$	15011
261	2358	$105,572$	$17830,8$
106	2547	$0,389$	$5805,89$
101	2452	$598,368$	$7192,73$
24	2152	$0,534$	$20171,9$
25	2151	$5,826$	$22006,5$
107	2482	$0,365$	$6243,71$
100	2363	$27,792$	$6385,56$
109	2499	$0,349$	$5858,14$
100	2380	$28,327$	$6708,91$
195	5615	$1,603$	$70465,3$
187	5421	$436,25$	$76834,2$
354	6704	$8,815$	$12803,1$
312	6612	$1093,66$	$16202,9$

the chromatic number in graphs, we proved that CSP belongs to the class of $\mathcal{NP}$-Hard problems. As a consequence, we develop a heuristic resolution.
We proceed in two stages. The first one is Scheduling, where different services where assigned to atoms inspired in graph-coloring techniques. The second one is Contract. Here, we define the minimum staff for each atom by means of Monte Carlo simulation.
It is known from queuing theory that, under homogeneous Poisson arrivals and exponential services, the performance in terms of waiting times is better in one queue that in more. Following this hint, we remark that in the scheduling stage we stressed each atom, assigning the maximum number of compatible services. Finally, numerical results confirm this intuitive idea: it is better to build the minimum number of independent units, each one focused on compatible services. However, there are different colorings to achieve the chromatic number of a graph, and they lead to different staff levels.
As a future work, we would like to further analyze multidimensional arrivals and waiting times, queuing policies, introduce graph colorings with higher-performance and develop a full study on a real corporation.

ACKNOWLEDGMENT

Our most sincere gratitude to Dr. Claudio Risso and Dr. Pablo Rodríguez-Bocca for their meetings in order to develop the Contract-Scheduling Problem here introduced.

RECEIVED: JANUARY, 2014
REVISED: APRIL, 2015

REFERENCES

[1] ASMUSSEN, S. [2003]: **Applied Probability and Queues** Applications of mathematics (Springer).: Stochastic modelling and applied probability. Springer.

[2] CASTILLO, I., JORO, T., and LI, Y. Y. [2009]: Workforce scheduling with multiple objectives **European Journal of Operational Research**, 196, 1,:162 – 170.

[3] CHANNOUF, N. and L'ECUYER, P. [2012]: A normal copula model for the arrival process in a call center **International Transactions in Operational Research**, 19, 6,:771–787.

[4] COOK, S. A. [1971]: The complexity of theorem-proving procedures In **Proceedings of the third annual ACM symposium on Theory of computing**, STOC '71, pages 151–158, New York, NY, USA. ACM.

[5] FOWLER, J. W., WIROJANAGUD, P., and GEL, E. S. [2008]: Heuristics for workforce planning with worker differences **European Journal of Operational Research**, 190, 3,:724 – 740.

[6] GAREY, M. R. and JOHNSON, D. S. [1979]: **Computers and Intractability: A Guide to the Theory of NP-Completeness** W. H. Freeman and Company, New York, NY, USA.

[7] GRAHAM, R. and GRÖTSCHEL, M. [1995]: **Handbook of combinatorics. 1 (1995)** Handbook of Combinatorics. Elsevier.

[8] Karp, R. M. [1972]: Reducibility among combinatorial problems In Miller, R. E. and Thatcher, J. W., editors, **Complexity of Computer Computations**, pages 85–103. Plenum Press.

[9] LITTLE, J. D. C. [1961]: A proof for the queuing formula: $l = \lambda w$ **Operations Research**, 9, 3,:383–387.

[10] LOTFI, V. and SARIN, S. [1986]: A graph coloring algorithm for large scale scheduling problems. **Computers and Operations Research**, 13, 1,:27–32.

[11] MALAGUTI, E. and TOTH, P. A survey on vertex coloring problems **International transactions in operational research (ITOR)**, 17.

[12] SAVINO, M. M., BRUN, A., and MAZZA, A. [2014]: Dynamic workforce allocation in a constrained flow shop with multi-agent system **Computers in Industry**, 65, 6,:967 – 975.

[13] STIRPE, L., BONACHE, J., and REVILLA, A. [2014]: Differentiating the workforce: The performance effects of using contingent labor in a context of high-performance work systems **Journal of Business Research**, 67, 7,:1334 – 1341.

[14] THOMSON, F. [1979]: A graph coloring algorithm for large scale scheduling problems. **Journal of Research of the National Bureau of Standars**, 84, 6,:489–506.

[15] WU, Q. and HAO, J. K. [2012]: Coloring large graphs based on independent set extraction **Comput. Oper. Res.**, 39:283–290.