Proceedings of the 5th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed Systems, Madrid, Spain, February 15-17, 2006 (pp1-7)

Cost Minimization in the Design of IT Infrastructures

Danilo Ardagna, Chiara Francalanci
Dipartimento di Elettronica e Informazione
Politecnico di Milano, Milan, Italy

Marco Trubian
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano, Milan, Italy

Abstract: The selection of a cost-minimizing combination of hardware and network components that satisfy requirements is a complex design problem with multiple degrees of freedom. Decisions must be made on how to distribute the computing load onto multiple computers and where to locate computers. This paper provides an overall methodology for combining hardware and network design in a single cost-minimization problem for multi-site computer systems. Costs are minimized by applying a heuristic optimization approach to a sound decomposition of the problem. Verifications consider several test cases with different computing and communication requirements. Cost reductions are evaluated by comparing the cost of methodological results with those of architectural solutions obtained by applying professional design guidelines.

Key-Words: capacity planning; cost minimization; optimization

1 Introduction

The information technology (IT) infrastructure is comprised of the hardware and network components of a computer system [13]. The objective of infrastructural design is the minimization of the costs required to satisfy the computing and communication requirements of a given group of users [12]. In most cases, multiple combinations of infrastructural components can satisfy requirements and, accordingly, overall performance requirements can be differently translated into processing and communication capabilities of individual components. These degrees of freedom generate two infrastructural design steps: the selection of a combination of hardware and network components and their individual sizing (see Figure 1). Cost-performance analyses are executed at both steps. Performance analyses receive a pre-defined combination of components as input and initially focus on the application of mathematical models to define the configuration of each component [13]. Performance bottlenecks are then identified at a system level and removed by re-sizing specific components that constrain system-level performance. Conversely cost analyses start at a system level, to identify a combination of components that minimizes overall costs, which is initially calculated from rough estimates of individual components' configurations and corresponding costs. The evaluation of costs of individual components is subsequently refined based on more precise sizing information from performance analyses (see Figure 1). Due to this interdependence between cost and performance analyses at both design steps, the overall infrastructural design process is iterative.

The goal of this paper is to support the cost-oriented design of modern IT infrastructures with an approach based on mathematical programming tools. Infrastructural design alternatives are organized within a methodological framework and are provided a formal representation suitable for optimization. Optimization is accomplished by sequentially solving two set-partitionings problems, a min k-cut problem with a non linear objective function and tuning the given solution with a tabu-search heuristic. The candidate minimum cost infrastructure are meant to be further analyzed by applying fine-tuning performance evaluation techniques.

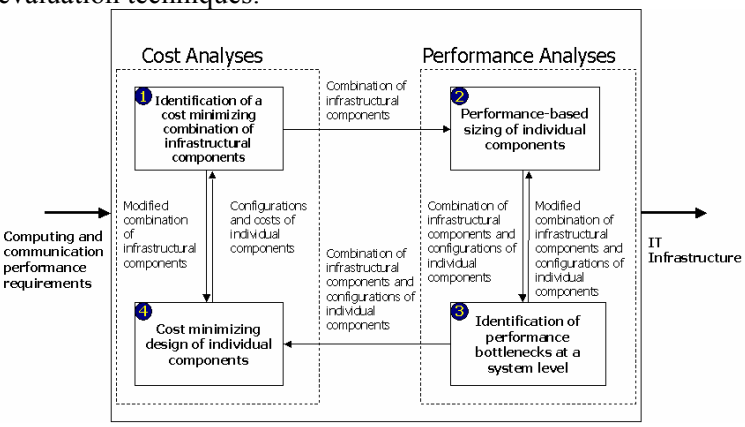

Figure 1 - The infrastructural design process.

Current professional guidelines generally recommend solutions to individual design alternatives that translate into an overall centralization of hardware components [6; 9]. However, only a few academic studies have attempted a more systematic analysis of cost issues in infrastructural design [12]. Hence, the goal of this paper is also to verify current professional design guidelines suggesting centralization as a general paradigm for cost minimization.

The next section presents the optimization problems which are modelled in Section 3. Section 4 describes the algorithmic approach proposed to solve the optimization models. Section 5 discusses the results of the empirical verification of the approach and the professional design guidelines. Conclusions are drawn in Section 6.

2 The optimization problem

In the following we assume that WANs are implemented as IP Virtual Private Networks (VPNs) [15], due to their flexibility in realizing point-to-point connections. In this way, network design is performed by sizing link capacity and calculating the associated TCO. The next section formalizes the technology requirements of an organization. The reference organization has a set S of sites.

2.1 Technology requirements

Definition 1. *Sites.* A site denoted by $s \in S$ is defined as a set of organizational resources connected by a Local Area Network (LAN).

Definition 2. *Instances of server applications.* An instance of a server application (or application process, or, simply, application) $a_i \in A$ is characterized by: (a) the operating system, O_i, (b) the computing capacity requirements, indicated as $Mips_i$ and measured in millions of instructions per second and (c) primary memory requirements, indicated as Ram_i and measured in mega bytes.

Definition 3. *User classes.* Each site $s \in S$ has a set of user classes C^s. Each user class $c_i^s \in C^s$ is a set of users with common computing requirements, that is, using the same set of applications $A_i \subset A$. A user class is also characterized by a *type* of client computers, either *fat*, *thin* or *hybrid*. The client type assigned to a user class is not a design alternative. Client machines are located at the same site as their user class. If *type* is *thin* or *hybrid*, the following additional characteristics are defined: (a) the primary memory size, Ram_i, measured in mega bytes and (b) the computing capacity, $Mips_i$, required to support the remote execution of applications for the whole class. C is defined as the set of all user classes, that is $C = \bigcup_{s \in S} C^s$.

2.2 Technology resources

The computing requirements of the reference organization can be satisfied by means of the following hardware resources: servers and thin servers. Each hardware resource is characterized by its configuration.

Definition 4. *Configuration.* A configuration indexed by $k \in SC$ is characterized by the following parameters: (a) the primary memory size, RAM_S_k, measured in mega bytes and (b) the computing capacity, $MIPS\_S_k$, measured in millions of instructions per second.

Definition 5. *Server.* A server is a computer with configuration $k \in SC$ that supports application instances.

Definition 6. *Thin server.* A thin server is a computer with configuration $k \in SC$ that supports thin or hybrid client computers.

Definition 7. *Cluster.* A cluster indexed by $j \in CL$ is defined as a set of either servers or thin servers characterized by the same configuration. The configuration of the server machines composing a cluster is also referred to as cluster configuration. Thin servers and servers cannot coexist in the same cluster. A cluster that includes thin servers is also referred to as *thin cluster*. N represents the maximum number of servers in a cluster; the actual number of servers in cluster j is $n(j) \le N$.

Definition 8. *Network connection.* A network connection is a dedicated communication link n_s connecting site $s \in S$ to the VPN. A network connection is bi-directional and its capacity BS_s is defined as an ordered pair (BO_s, BI_s), where BO_s and BI_s represent input and output capacity, respectively, and are both measured in [bit · s^{-1}].

Definition 9. *Data exchanges.* User classes and applications exchange data. Data exchanges are modelled as a directed weighted graph $G=(V,E)$. A vertex $v \in V$ of graph G can represent an application, a user class or a thin server. E is defined as the set of all arcs of graph G. The weight $R_{\alpha\beta}$ associated with the directed arc $(\alpha,\beta) \in E$ connecting a generic vertex $\alpha \in V$ to a generic vertex $\beta \in V$, represents the average bandwidth required to support the data exchanges from vertex α to vertex β. In general, $R_{\alpha\beta}$ is a function of the frequency of data exchanges and their average size. A special node $v_0 \in V$ represents external applications exchanging data with internal applications through the Internet.

Constraint 1. *Sharing of clusters among user classes.* Not all user classes can share the same thin cluster. This is specified by defining groups $G_h^1 = \left\{ C_i^s \right\}$, with $h \in IG_1$, such that $\forall l \neq m$, $G_l^1 \cap G_m^1 = \varnothing$. Each group is a set of user classes that can share the same thin cluster.

Constraint 2. *Sharing of clusters among applications.* Not all server applications can share the same cluster. This is specified by defining groups $G_h^2 = \{a_i\}$, with $h \in IG_2$, that is sets of server applications a_i that can share the same cluster.

Observations: note that IG_1 identifies a partition, since user classes are usually partitioned for security reasons or privileges. On the other hand, the groups identified by IG_2 can overlap. In this way, multiple allocations can be defined for server applications. MIPS requirements for application instances are evaluated by considering requests throughput and demanding time [13]. Application instances will be assigned to clusters whose overall capacity is greater than or equal to their MIPS requirements. This guarantees maximum CPU utilization. In this paper, the MIPS of server machines are estimated in such a way that maximum CPU utilization is lower than 60% [1,2,4]. With values of utilization greater that 60%, small variations of throughput would cause a substantial growth of response time and, overall, performance would become unreliable. This empirical rule of thumb, which is commonly applied in practice [13], has been provided a formal validation. It has been formally demonstrated that a group of aperiodic tasks will always meet their deadlines as long as the bottleneck resource utilization is lower than 58% [2]. Note that performance analyses should follow cost analyses to refine sizing according to a formal queuing model. The aim of our work is to find a candidate minimum-cost infrastructure that can be fine-tuned by applying performance evaluation

techniques. Note that MIPS are evaluated for homogeneous classes of servers (for example, Intel machines will not be compared with SPARCs). Therefore, each application will be allocated on a specific class of servers. Similarly, the primary memory of each server in a cluster should be grater than or equal to the summation of the RAM requirements of all applications a_i simultaneously executed by the cluster. For the sake of simplicity, server disk performance is not considered and it is assumed that server configurations are CPU and I/O balanced and that disks are never a bottleneck. The LAN connection equipment inside a site is not taken into account, as its cost per bit/s is several orders of magnitude smaller than the cost of leased network connections [15].

3 The optimization model

Let us enumerate from 1 to $|J|$ all subsets of user classes or server applications which are feasible according to the definition of groups G_h^1 and G_h^2. Let B denote the $|C \cup A| \times |J|$ matrix whose column B_j represents the characteristic vector of the j^{th} subset of client machines or server applications, defined as follows: the i^{th} entry b_{ij} of B_j is equal to one if the client machine or server application i belongs to the j^{th} subset, while it is equal to zero otherwise. Each column B_j is univocally associated with the minimum-cost cluster that can support all client machines or server applications in set j. Note that each set $j \in J$ either contains elements from C or it contains elements from A. Each set $j \in J$ is further constrained by the fact that only feasible allocations of user classes and server applications to clusters are allowed, according to groups $G_h^1 = \{C_i^s\}$, with $h \in IG_1$, and $G_h^2 = \{a_i\}$, with $h \in IG_2$ (see Constraints 1 and 2 in Section 2).

3.1 Decision variables

In our model optimization alternatives are represented by the following decision variables.

1. Selection of clusters:

$$x_j = \begin{cases} 1 & \text{if the } j\text{-th cluster in } J \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$

2. Allocation of clusters to sites:

$$y_j^s = \begin{cases} 1 & \text{if the } j\text{-th cluster in } J \text{ is allocated to site } s \in S \\ 0 & \text{otherwise} \end{cases}$$

$$w_{\alpha\beta} = \begin{cases} 1 & \text{if user class or server applications } \alpha \text{ and } \beta \text{ are} \\ & \text{allocated to clusters on different sites} \\ 0 & \text{otherwise} \end{cases}$$

3.2 Objective function

TCO, that is the objective function to be minimized, is defined as the summation of hardware investment costs, hardware management costs and the network costs.

1. *HW investment costs* = $\sum_{j \in J} c_j x_j$

Parameter c_j represents the cost of the minimum-cost cluster that can support all the user classes or server applications in set $j \in J$. That is to say, $c_j = \min_{k \in SC_j} \{n(j)(acq\_c_k + \sum_{a_i \in B_j} lic\_c_{ik})\}$, where SC_j denotes the subset of server configurations that can support the user classes or server applications in set j, $n(j)$ denotes the number of servers in cluster j, acq_c_k denotes the acquisition cost of servers with configuration k (including the installation and the operating system costs [4]), and lic_c_{ik} denotes the license cost of server application a_i when installed on configuration k (this term evaluates to zero when cluster j connects user classes, since the license cost of client applications remotely executed by a thin cluster only depends on the number of users [5]).

2. HW management costs = $\sum_{s \in S} \mathrm{Mng}_s(\sum_{i \in C^s} mng_i + \sum_{j \in J} y_j^s p_j)$

Parameter p_j represents the number of management hours required by cluster j (it depends both on the user classes or server applications allocated on cluster j and on cluster j's configuration, say $k \in SC$). I.e., $p_j = \sum_{i \in B_j} p_{ik}$, where parameter p_{ik} represents the number of management hours required by user class c_i^s or application a_i on cluster j with configuration k. Parameter mng_i indicates the number of management hours required by the client machines of user class c_i^s. Hardware management costs are computed as a non linear function, $Mng_s(\cdot)$. The argument of function $Mng_s(\cdot)$ is the total number of management hours required by all applications and user classes allocated on all clusters assigned to site s. Management hours can be either attributed to internal personnel or purchased. In the first case, they involve *in house* costs, which are a step-wise function of the number of people that must be hired to provide the required amount of management hours; in the second case, they involve *outsourcing* costs, which are a linear function of management hours. $Mng_s(\cdot)$ is calculated as the minimum value between the in house and outsourcing functions of costs.

3. Network costs = $\sum_{s \in S} TC_s(\sum_{(\alpha,\beta) \in E} \sum_{t \in S} R_{\alpha\beta} w_{\alpha\beta}, \sum_{(\alpha,\beta) \in E} \sum_{t \in S} R_{\alpha\beta} w_{\alpha\beta})$

Network costs of site s are computed as a two dimensional stepwise linear function, $TC_s(\cdot,\cdot)$, of the physical bandwidth required to support *inbound* and *outbound* information exchanges between user classes, applications or thin clusters of site s and any other site.

3.3 Problem formulation

The overall optimization problem is modelled as follows:

$$\text{P1)} \quad \min \sum_{j\in J} c_j x_j + \sum_{s\in S} \mathrm{Mng}_s\Big(\sum_{i\in C^s} mng_i + \sum_{j\in J} y_j^s p_j\Big) + $$

$$+ \sum_{s\in S} \mathrm{TC}_s\Big(\sum_{(\alpha,\beta)\in E}\sum_{t\in S} R_{\alpha\beta} w_{\alpha\beta},\ \sum_{(\alpha,\beta)\in E}\sum_{t\in S} R_{\alpha\beta} w_{\alpha\beta}\Big)$$

such that:

$$\sum_{j\in J} b_{ij} x_j = 1, \forall i \in C \cup A \qquad (1)$$

$$x_j - \sum_{s\in S} y_j^s = 0, \forall j \in J \qquad (2)$$

$$y_h^s - y_k^s \le w_{\alpha\beta}, \forall s \in S, \forall h,k \in J, h \ne k,$$
$$\alpha \in B_h, \beta \in B_k, \forall (\alpha,\beta) \in E \qquad (3)$$

$$x_j \in \{0,1\}, \forall j \in J;\ y_j^s \in \{0,1\}, \forall j \in J,$$
$$\forall s \in S;\ w_{\alpha\beta} \in \{0,1\},\ \forall (\alpha,\beta) \in E.$$

Constraint family (1) imposes that each user class or server application is assigned to exactly one cluster. Constraint family (1) summarizes two disjoint *set partitioning* problems: the first one selects thin clusters for user classes, while the second selects clusters for server applications. Constraint family (2) imposes that each cluster is assigned to exactly one site. It models the allocation of servers to sites. Constraint family (3) ties localization variables y to variables w. A variable $w_{\alpha\beta}$ must evaluate to one if user classes or applications α and β have not been assigned to clusters allocated in the same site s and exchange data with each other in graph G. Constraint families (2) and (3) together model the feasible region of a *min k cut* problem.

4 Cost optimization algorithm

The overall optimization problem has been split into the following three intertwined sub-problems, which are solved in sequence. A final fine-tuning step that implements a tabu-search approach is also performed, in order to improve the, possibly, local optimum that is found through the isolated solution of the four sub-problems. (1) *Client optimization*: user classes are assigned to minimum-cost thin clusters that satisfy requirements. (2) *Server optimization*: server applications are assigned to minimum-cost clusters of servers that satisfy computing requirements. (3) *Server localization*: the server machines identified by solving sub-problems (1) and (2) are allocated to sites by minimizing overall network and management costs.

Client optimization Disjoint sets of client computers that can share the same thin cluster, according to Definition 7, are assigned to the same thin cluster. This assignment is modeled as a family of *set partitioning problems* (SPPs), one for each group G_h^1, with $h \in IG_1$. Let us enumerate all the non empty subsets of elements in G_h^1 from 1 to $|J_h|$, for a given $h \in IG_1$. Let B denote the $|G_h^1| \times |J_h|$ matrix whose column B_j represents the characteristic vector of the j^{th} subset of user classes, Q_j (see Section 3). Each column B_j corresponds to a cluster that can support the overall Ram_i and $Mips_i$ requirements of all user classes in Q_j. A cost $c_j = \min_{k\in SC_j}\{n(j)\,acq\_c_k\}$ is associated with each column B_j and corresponds to the acquisition cost of the servers in the cluster (see Section 3.3), where SC_j denotes the subset of server configurations that can support the user classes in Q_j and $n(j)$ denotes the number of servers in the cluster made of servers with configuration $k \in SC_j$. Let x_j denote a binary variable which is equal to one if the j^{th} cluster in J_h is selected, zero otherwise. The optimization problem can be modeled as:

$$\text{P}_h) \quad \min \sum_{j\in J_h} c_j x_j$$
$$\sum_{j\in J_h} b_{ij} x_j = 1, \forall i \in G_h^1 \qquad (4)$$
$$x_j \in \{0,1\}, \forall j \in J_h$$

Each feasible solution identifies a set of clusters such that each user class in G_h^1 is connected to exactly one of them.

Server optimization This sub-problem considers the optimum allocation of server applications to clusters. The set of servers involved in this sub-problem excludes thin/hybrid servers. Server applications are organized in tiers, i.e. into sets of server applications that cooperate to manage the same request. Each server application or application tier must be assigned to exactly one cluster. Similar to the client optimization problem, this problem is modeled as a SPP. Let us enumerate from 1 to $|J|$ all subsets of elements in G_h^2, for all $h \in IG_2$. Let B denote the $|A| \times |J|$ matrix whose column B_j represents the characteristic vector of the j^{th} subset of server applications, Q_j. Each column B_j corresponds to a cluster such that each *individual* server in the cluster has enough *memory* to support all server applications in Q_j. Similarly, the number of servers in the cluster provides enough computing capacity to support all server applications in Q_j. A cost c_j, corresponding to the acquisition cost of all servers in the cluster (see Section 3.3) is associated with each column B_j. Let x_j denote a binary variable which evaluates to one if the j-th cluster in J is selected, to zero otherwise. The optimization problem can be modeled as:

$$\text{P2)} \quad \min \sum_{j\in J} c_j x_j$$
$$\sum_{j\in J} b_{ij} x_j = 1,\ \forall i \in A \qquad (5)$$
$$x_j \in \{0,1\}, \forall j \in J_h$$

Each feasible solution of the SPP P2) identifies a set of clusters such that each server application in A is allocated to exactly one cluster of the identified set.

Server localization This sub-problem considers the optimum allocation of clusters to sites. Two cost items are affected by the allocation of clusters: hardware management and network costs which are evaluated by means of the $Mng_s(\cdot)$ and $TC_s(\cdot,\cdot)$ functions, respectively (see Section 3.3). This cost-minimization sub-problem can be modelled as a network optimization problem as follows. Let us consider a directed graph $G = (V, E)$ and a subset $V_{CL} \subseteq V - \{v_0\}$. Vertices in V_{CL} represent clusters, while vertices in $V - V_{CL} - \{v_0\}$ represent client computers which are located at the same site as their user class. The set of arcs E represents possible data exchanges between client computers and clusters, among clusters and between v_0 (i.e., the special node which represents external applications) and clusters. The problem consists in partitioning V_{CL} into disjoint subsets J_s, with $s = 1,\ldots, |S|$, in order to minimize the following objective function:

$$\sum_{s\in S}(\mathrm{Mng}_s(\sum_{i\in C^s} mng_i + \sum_{j\in J_s} p_j) + \mathrm{TC}_s(\sum_{\substack{(\alpha,\beta)\in E\\ \alpha \text{ in } s\\ \beta \text{ in } t\neq s}} R_{\alpha\beta} + \sum_{\substack{(\alpha,v_0)\in E\\ \alpha \text{ in } s}} R_{\alpha v_0}, \sum_{\substack{(\alpha,\beta)\in E\\ \beta \text{ in } s\\ \alpha \text{ in } t\neq s}} R_{\alpha\beta} + \sum_{\substack{(v_0,\beta)\in E\\ \beta \text{ in } s}} R_{v_0\beta}))$$

where α (β) in s (t) denotes that vertex α (β) has been located at site s (t). For each site s, the first term is management cost of all clusters and user classes located in s, and the second term represents the cost of the bandwidth required to connect vertices in s with vertices in sites different from s. In each feasible solution, each cluster is assigned to one site and each set of client computers is assigned to the site of the corresponding user class. If all values p_i evaluate to zero, V_{CL} is equal to V and network costs are a linear function of bandwidth, the above network problem is known in literature as *min k-cut* problem, where $k = |S|$. A tabu search meta-heuristic [9] has been adopted. The neighborhood of each feasible solution is defined by all solutions that can be obtained by moving a cluster to a different site, for all clusters. Only the short-term memory mechanism has been implemented.

Fine-tuning step The decomposition of the overall optimization problem into four sub-problems does not guarantee that the final solution is a global optimum. Hence, a fine-tuning step based on a tabu-search approach [9] has been implemented to possibly improve the solution obtained by separately solving the four sub-problems. Only the short-term memory mechanism has been implemented. The neighborhood of a solution is defined as follows. A user class, say C_i^{s1}, or a server application, say a_j, is disconnected from a cluster, say $\mathrm{Cluster_A}$, to which it is currently connected. A new minimum-cost cluster, say $\mathrm{Cluster_B}$, is selected to replace $\mathrm{Cluster_A}$. Costs are evaluated by assuming that $\mathrm{Cluster_B}$ is located in the same site, say s_2, of the cluster that is replaced. A new minimum-cost cluster, say $\mathrm{Cluster_C}$, is selected to support C_i^{s1} (or a_i) and the costs of allocating $\mathrm{Cluster_C}$ in a site different from s_2 are evaluated. Hardware management costs and network costs are calculated by means of the $Mng_s(\cdot)$ and $TC_s(\cdot,\cdot)$ functions. In this way, a destination site, say s_3, is identified for $\mathrm{Cluster_C}$. At last, the possibility of discarding $\mathrm{Cluster_C}$ is evaluated by connecting C_i^{s1} (or a_i) to a different cluster in s_3.

5 Empirical verifications

Empirical verifications have been supported by ISIDE (Infrastructure Systems Integrated Design Environment), a prototype tool that implements the cost minimization algorithm. The tool includes a database of commercial infrastructural components and related cost data described in [3] which includes about 5000 server configurations from 4 vendors. For the solution of linear integer programming models ISIDE calls CPLEX 8.0 library routines. Simulations have been supported by a PIV@3GHz, Windows XP workstation with 1 GB of RAM. Analyses focus on three case studies: a multi-department university, an Internet banking system and an information retrieval system. The three case studies have substantially different technology requirements. In the first case study, user classes are numerous and use a variety of applications, making the allocation of servers to sites a critical design alternative. The Internet banking system is composed of complex multi-tier applications whose allocation on servers is particularly cumbersome. Finally, the information retrieval system is characterized by CPU-intensive applications and the design of server farms plays an important role. The computing requirements data of the test cases are reported in [3]. In order to evaluate the performance of the cost-minimization algorithm, each case study is analyzed for an increasing number of user classes, applications and sites. Cost and time efficiency are evaluated by comparing the algorithm's output with the output of the fine-tuning step starting from an initial solution obtained by applying the following rules:
(1) User classes adopting thin or hybrid client computers and belonging to the same group G_h^1 are assigned to a single cluster, according to the server-consolidation principle [9,11]. (2) Applications belonging to the same group G_h^2 are assigned to a single cluster, according to the server-consolidation principle [9,11]. Applications belonging to multiple groups are allocated to the cluster that maximizes the number of tiers of requests [14]. (3) All servers are located in one site, which is selected by minimizing management costs, according to the server consolidation principle [9]. (4) Clusters are implemented by selecting the smallest server that can support applications, to reduce hardware acquisition costs, according to the "think big, but build small" paradigm [14].
In the following, the final algorithm's solution will be indicated as $\mathrm{Sol_A}$. The solution identified by applying rules 1-5 will be referred to as $\mathrm{Sol_B}$, while the final solution obtained by applying the fine-tuning step to $\mathrm{Sol_B}$ will be indicated as $\mathrm{Sol_C}$.

- *Improvement of the professional initial solution* (IPI): it represents the percent improvement of $\mathrm{Sol_B}$ and is a measure of the efficiency of our optimization approach

compared to professional design guidelines. It is evaluated as $(Sol_B - Sol_A)/Sol_A$;

- *Improvement of the professional final solution* (IPF): it represents the percent improvement of Sol_A with respect to Sol_C. It is evaluated as $(Sol_C - Sol_A)/Sol_A$.

5.1 A multi-department university

The university is composed by K departments (sites), with K ranging between 1 and 6. Each site hosts 3 user classes, administrative staff, software engineering researchers (SE) and electronic engineering researchers (EE). Each class is composed by 100 users. All users use a browser, an e-mail client and an office-automation suite. SE researchers also use an integrated development environment and EE researchers also use a circuit simulator. The administrative staff is assigned to thin clients, while researchers are assigned to hybrid clients. An e-mail and a web/proxy server application are introduced for each department, running on W2000 and Linux servers, respectively. The web server is accessed by internal clients but also by external Internet users. Data on RAM, MIPS and data exchanges among server applications have been empirically obtained from the analysis of our University's system logs. Two groups are specified, G_1^2 for e-mail servers and G_2^2 for web servers. A single group G_1^1 including all user classes is specified. The solution identified by our methodology is fully distributed for all values of K. For each site, e-mail and web applications are allocated on dedicated servers and a thin cluster supporting all user classes is introduced. If Sol_A is compared with Sol_B, cost savings are higher than 280%, since professional guidelines suggest the centralization of servers in a single site. For the one-site test case, the professional solution is improved by about 20%, since in this case the difference between the methodological and professional solution is only due to a different sizing of servers. Table 1 summarizes the total execution time and the metrics defined above, as a function of the number of sites K. In general, as the size of the system (i.e., the number of sites) grows, the improvement of both the initial and final professional solutions increases.

N. of sites	Time	IPI	IPF
K=1	2.2 s	20.31%	18.22%
K=2	3.4 s	287.10%	281.76%
K=3	9.8 s	316.86%	302.88%
K=4	15.4 s	359.93%	346.46%
K=5	30.1 s	373.89%	348.83%
K=6	43.3 s	384.39%	366.14%

Table 1 - Summary of results of a multi-department university.

5.2 An Internet banking system

The system is distributed over K sites, with K ranging between 1 and 3. Each site supports 100.000 Internet users accessing the following applications: (a) a web server application; (b) a servlet engine; (c) an application server; (d) a relational DBMS storing historical data on stock quotes; (e) an object-oriented DBMS storing user data. Users issue two types of requests, information retrieval and transaction execution, with a 10 to 1 ratio. The overall average access rate to the system is about 250 accesses per hour. Data on user classes, RAM, MIPS and data exchange requirements among server applications have been obtained from the logs of a large national financial institution. The target system for the optimization is based on Ultra Sparc Solaris servers (see the observation on MIPS in Section 2). DBMSs are replicated in all sites and transactions write multiple copies of data synchronously for fault tolerance purposes. Three different allocations of applications into tiers are allowed: a 5-tier allocation, which assigns each server application to a single tier, and two 4-tier allocations, which assign the servlet engine to the same tier of either the web server or the application server. This is obtained by introducing two G_h^2 groups: the first group includes web servers and servlet engines; the second group includes servlet engines and application servers. Results are reported in Table 2. The professional solutions are not improved by the final fine-tuning step. The decomposition is effective, as it enables a 10-20% reduction of TCO, which increases with the size and complexity of the system. Sol_A and Sol_C are different from each other. In Sol_A, database servers are replicated in all sites, according to design constraints, and web applications are centralized on one cluster in one site. Servlet engines and application servers are allocated on the same cluster, but one such cluster is allocated on each site to serve the site's user classes. This contrasts against professional guidelines suggesting the allocation of applications with the maximum number of tiers and the centralization of corresponding tiers of different instances of the same application. Sol_C allocates all server applications to the same site on 5 tiers.

N. of sites	Time	IPI	IPF
K=1	1.3 s	10.52%	10.52%
K=2	8.2 s	15.06%	15.06%
K=3	19.1 s	19.79%	19.79%

Table 2 - Summary of results of an Internet banking system.

5.3 An information retrieval system

The following test case is based on the work presented in [7]. An information retrieval system is considered, composed by two applications: (a) the information retrieval engine, called *inquery server*, which stores and retrieves documents by providing a query interface; (b) the central broker, called *connection server*, which administers the connection between users and inquery servers by maintaining a list of available connections and by routing queries and responses, accordingly. The system is distributed over K sites, with K ranging between 1 and 3.

Each site has a local connection server and four inquery servers. Local connection servers distribute requests to both local and remote inquery servers (which store different data). External users access the distributed system from the Internet and can issue search retrieval and document retrieval commands. Requests are uniformly distributed across inquery servers. The target system for the optimization is based on Alpha servers. Data on MIPS and RAM requirements and data exchanges are obtained from [7]. Two groups are introduced, G_1^2 including all the instances of connection servers and G_2^2 including all inquery servers. In both Sol_A and Sol_B severs are localized in one site. Anyway, in Sol_A, connection servers are centralized within the same cluster, while each instance of the inquery server is assigned to a separate cluster. The professional solution suggests the location of all servers in one site and the centralization of both connection and inquery servers in two clusters. The fine-tuning step improves Sol_B by 2-4% (see Table 3). The decomposition is effective as it identifies a solution 20-55% cheaper than Sol_C. Sol_A is also 30-60% cheaper than Sol_B.

N. of sites	Time	IPI	IPF
K=1	1.5 s	28.44%	23.88%
K=2	2.8 s	49.36%	46.46%
K=3	177.98 s	64.31%	58.19%

Table 3 - Summary of results of an information retrieval system.

6 Conclusions

We have proposed an overall approach to support the cost-oriented design of hardware and network systems considering both infrastructural, network and management costs. Most of the design alternatives enabled by current technologies are addressed, including server sizing and localization of multi-tier applications. Cost reductions have been evaluated by comparing the cost of methodological results with those of architectural solutions obtained by applying professional design guidelines. Testing results indicate that cost reductions can be significant and considerably grow with the size and complexity of the system. Current professional rules are challenged by findings, this indicates that general design guidelines are difficult to infer. Analyses show that the geographical centralization of servers can reduce management costs, but cannot be assumed as a universal cost-minimizing paradigm [10,11]. Similarly, a higher number of tiers does not represent a reliable source of savings [14]. From a methodological standpoint, cost-oriented design is suitable for the selection of a combination of technology resources, while it involves an approximation in their individual sizing. Future work will consider the integration of the cost-oriented algorithm with traditional performance analyses which provide precise sizing information.

References:

[1] T.F. Abdelzaher, K.G. Shin, N. Bhatti, "User-level QoS-adaptive resource management in server end-systems," IEEE Trans. on Computers, 52, 678-685, 2003.

[2] T.F. Abdelzaher, K.G. Shin, N. Bhatti, "Performance Guarantees for Web Server End-Systems: A Control-Theoretical Approach," IEEE Trans. on Parallel and Distrib. Systems, 13, 80-96, 2002.

[3] D. Ardagna, "A cost-oriented methodology for the design of information technology architectures", Politecnico di Milano, Ph. Dissertation, http://www.elet.polimi.it/upload/ardagna/phd_dissertation.pdf, 2004

[4] D. Ardagna, C. Francalanci, M. Trubian, "A cost-oriented approach for infrastructural design," ACM SAC2004 Proc., 2004.

[5] D. Ardagna, C. Francalanci, "A Cost-Oriented Approach for the Design of IT Architectures", Journal of Information Technology 20, 32-51, 2005.

[6] D. Ardagna, C. Francalanci, G. Bazzigaluppi, M. Gatti, F. Silveri, M.Trubian, "A Cost-oriented tool to support server consolidation", ICEIS 2005 Proc., 2005.

[7] B. Cahoon, K. S. McKinley, Z. Lu, "Evaluating the performance of distributed architectures for information retrieval using a variety of loads," ACM Trans. on Information Systems, 18, 1-43, 2000.

[8] Gavish, B., Pirkul, H, " Computer and Database Location in Distributed Computer Systems," IEEE Trans. on Computers. 35(7), 583-590, 1986.

[9] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publishers, 1997.

[10] HP, "The Scope of HP Systems Consolidation," http://www.hp.com/products1/unixservers/solutions, 2002.

[11] IBM, "Improve the Return on Your IT Investment with Server Consolidation," http://www-1.ibm.com/servers/eserver/iseries/australia/serv.htm, 2002.

[12] H.K. Jain, "A Comprehensive Model for the Design of Distributed Computer Systems," IEEE Trans. on Software Engineering, 13, 1092-1104, 1987.

[13] D.A. Menascé, and V.A.F. Almeida, "Scaling for E-business. Technologies, models, performance and capacity planning", Prentice-Hall, 2000.

[14] P. Sonderegger, H. Manning, "Best Practices For Web Site Reviews," Forrest Research, 2002.

[15] R. Yuan, W.T. Strayer, "Virtual Private Networks: Technologies and Solutions," Addison Wesley, 2001.