Dynamic QoS-Aware Coalition Formation

Luís Nogueira, Luís Miguel Pinho
IPP Hurray Research Group
Polythecnic Institute of Porto, Portugal
{luis,lpinho}@dei.isep.ipp.pt

Abstract

Users of wireless devices increasingly demand access to multimedia content with specific quality of service requirements. However, different users might tolerate different levels of service, or could be satisfied with different quality combinations choices. We provide semantically rich QoS specification interface for multidimensional QoS provisioning, allowing the user, and applications, to define fine-grained service requests. Traditionally, multimedia processing introduces heavy resource requirements on the client side. Our work tries to address this growing demand on resources and performance requirements, by allowing wireless nodes to cooperate with each other, or with wired neighbor nodes, to meet resource allocation requests and handle stringent constraints, opportunistically taking advantage of local network resources and processing power, as nodes move in range of each other, forming a temporary coalition for service execution. We argue that coalition formation is necessary when a single node cannot execute a specific service, and it may also be beneficial when groups perform more efficiently when compared to a single's node performance.

Keywords: Dynamic Real-time Systems, Quality of Service, Resource Management, Distributed Systems, Ad-hoc Networks

1. Introduction

Quality of Service (QoS) is considered an important user demand, receiving wide attention in real-time multimedia research[22]. However, in most systems, users do not have any real influence over the QoS they can obtain, since service characteristics are fixed when the systems are initiated. Furthermore, multimedia applications (and their users) can differ enormously in their service requirements as well as in the resources which need to be available to them[3]. Therefore, there is an increasing need for customizable services that can be tailored to user's specific requirements.

In the meantime, new multimedia applications present increasingly complex demands on quality of service, reflected in multiple attributes over multiple quality dimensions. Also, the use of laptop computers and handheld devices coupled with wireless network interfaces is growing rapidly. Such wireless connectivity to a local wired network makes the services available on that network to the mobile devices, creating a new, highly dynamic and decentralized environment for multimedia applications.

Such an environment is expected to be heterogeneous, consisting of nodes with several resource capabilities. For some of those there may be a constraint on the type and size of applications they can execute with user's acceptable quality of service. For example, video conferencing systems often use compression schemes that are effective, but computationally intensive, trading CPU time for limited network bandwidth. A mobile client with limited CPU and memory capacity, but sufficient link speed, with nearby more powerful (or less congested) devices, can divide the computational intensive processing into tasks and spread it among different neighbors.

Therefore, in this paper we address distributed multimedia processing on wireless devices in scenarios where multimedia data is heavily processed when transferred between a server and the mobile device, maximizing user's quality requirements on several dimensions. We are interested in supporting spontaneous and opportunistic behavior in this new dynamic environment by enabling cooperation between available nodes. The purpose of service allocation to a group of nodes is to maximize the benefits associated to a cooperative execution of services, addressing the increasing demands on resources and performance. Nodes may cooperate either because they can not deal alone with resource allocation demands or because they can reduce the associated cost by working together.

We seek a generic model that enables a distributed service allocation, *i.e.*, without a central authority distributing the services among nodes. Given a set of services, a distributed environment must seek the maximisation of the associated QoS constraints. The nodes shall reach efficient

service allocation by themselves, seeking a maximal outcome. This will be achieved via the formation of a temporary group of individual nodes, which, due to its higher flexibility and agility, is capable of effectively respond to new, challenging, requirements. We call these groups *coalitions*.

The basic components of such a smart environment will be nodes able to organize flexibly into a coalition for service processing. It is clear that such a group presents very significant challenges, especially at the architectural level. Major developments are required in the fields of communications protocols, data processing and application support. Our goal is to develop the architecture which enables the creation of a new generation of nodes that can effectively network together, providing a flexible platform for the support of distinct network applications.

The rest of this paper is structured as follows. The next section provides a brief description of the considered model for the system. Afterwards, section 3 presents our approach for QoS requirements representation and service requests. Section 4 presents the service allocation and coalition formation processes, while sections 5 and 6 present, respectively, the proposals formulation and evaluation. Finally, section 7 presents some conclusions.

2. System model

Before building a model for this collaborative distributed processing, one must address the question if the integration is feasible and worth the effort.

Mobile devices are gaining popularity, and it is expected that the number of mobile devices will grow even more in the next few years. Very different types of mobile devices are currently available: telephones, PDAs, laptops, etc. Even the more simple ones are expected to have sooner or later sufficient processing resources to be used as multimedia clients.

At first glance, an individual mobile device may not have sufficient capacity and computation power for an effective integration in a distributed multimedia processing environment. However, if we exploit the *aggregated mobile power* instead of single, individual power and consider the exponential rise of mobile devices and the continuous developments in wireless technology, then one may conclude that this collaborative processing can be a valid solution.

The challenges introduced by these new requirements are already being addressed. Resource allocation in heterogeneous wireless networks has been previously addressed in [8]. The basic concept is that each service is delivered via the network that is most efficient to support the service. In [10] the authors introduced a fault tolerance architecture to provide continuous QoS support in case of network failures, by allowing users to access one of several wireless networks. Also, computation offloading on mobile computers has been previously explored [12][15][17][16]. In [15] the authors use offloading computation techniques for reducing execution time and energy consumption on handheld devices, assuming both input and output are local. A task partition/allocation scheme is proposed to allow the computation to be offloaded from the handheld to a server through a wireless LAN.

In this paper we consider a system where wireless/mobile nodes may dynamically enter the range of each other, and of wired infrastructures (even clusters of nodes[7]), oportunistacally taking advantage of local resources, distributed across neighbor nodes, forming temporary coalitions for service execution, considering QoS-aware applications.

Central to the behaviour of our framework is the *QoS Provider*, which is responsible for processing both local and distributed resource requests on each node. Rather than reserving resources directly, contacts the *Resource Managers* to grant specific resource amounts to the requesting task. We view a *Resource* as a limited hardware or software quantity supplied by a specific node. These might be CPU time, memory, I/O bus bandwidth, network bandwidth.

Each *Resource Manager* is an object that manages a particular resource. This typically would be implemented by the device driver that manages the physical resource, by the scheduler that manages the CPU, or by software that manages other resources (such as memory).

Within the QoS Provider, the *Negotiation Organizer* is responsible for the coalition formation process described in section 4.2, atomically distributing service requests, receiving individual nodes' proposals and deciding which node(s) will provide the service. We consider the existence of an atomic broadcast mechanism[1] in the system, guaranteeing that all nodes receive the same service requests and proposals in the same order. Also within the QoS Provider, the *Local Provider* is responsible for replying to negotiation requests and for maintaining the state of node's resource allocations and services provided.

3. QoS support in distributed multimedia systems

Over the last years there has been a considerable amount of research within the field of QoS support for distributed multimedia systems [4]. There is a lot of research on end-to-end architectures for QoS support [9][6][21][11] and on link, network and transport layers [23][20]. Some of this research has been focused in low-level systems' mechanisms. While these issues are extremely important factors for QoS control, they are not sufficient for end users. Users may require different levels of service, or could be satisfied with different quality combination choices [13].

Research on adaptive QoS control takes user's perspective into account by providing mechanisms for an application to adapt itself to changes in the environment, but the user has limited influence over the QoS delivered by the application. In [2] the authors propose a mechanism for QoS negotiation as a way to ensure graceful degradation. They suggest that a user should be able to express, in the service request, the spectrum of acceptable QoS levels, as well as the perceived utility of receiving service at each of those levels. However, these levels are statically mapped to certain quality choice combinations.

Research in QoS guarantees in wireless networks has been extensively studied in recent years. In [24] the authors classify the several proposed QoS schemes in three categories:

- Link adaptation in the physical layer
- Channel access coordination in the MAC layer
- Admission control strategies in MAC and higher layers

Essentially, the proposed approaches focus on different network layers and are tightly interrelated. Guaranteeing QoS in wireless networks is still a very challenging problem and needs further study [14].

3.1. QoS requirements specification

It is important to provide a semantically rich QoS specification that can be used by both the user and the system. Because QoS is often multi-dimensional the user (or application) might want to make some quality tradeoff, especially when available resources are scarce. Therefore, it is to the user's advantage to be able to specify his QoS requirements using an interface that allows him to make implicit or explicit quality tradeoffs.

We propose to describe those requirements through a scheme that defines dimensions, attributes and values, as well as relations that maps dimensions to attributes and attributes to values.

$$QoS = \{Dim, Atr, Val, DA_r, AV_r, Deps\}$$

where Dim is the set of QoS dimensions identifiers, $Attr$ is the set of attributes identifiers and Val is the set of attribute's values identifiers.

Values are represented by the following structure:

$$\begin{aligned} Val &= \{Type, Domain\} \\ Type &= \{integer, float, string\} \\ Domain &= \{continuous, discrete\} \end{aligned}$$

The relationship that assigns to each dimension in Dim a set of attributes in $Attr$ is defined as

$$DA_r : Dim_i \rightarrow Atr, \forall_{Dim_i} \in Dim$$

The relationship that assigns to each attribute in $Attr$ a specific value in Val is defined as

$$AV_r : Atr_i \rightarrow Val_k, \forall_{Atr_i} \in Atr, \exists^1{}_{Val_k} \in Val$$

The set of relationships defining the dependencies between attributes' values is defined as

$$Deps : \{Dep_{ij}\}$$

$$Dep_{ij} = f(Val_{ki}, Val_{kj}), \forall Attr_i, Attr_j \in Attr$$

Using a video streaming application as an example, we may define a set of dimensions (and their attributes) that might be associated with any particular application (the following list is not intended to be exhaustive).

$$\begin{aligned} Dim &= \{Video\ Quality,\ Audio\ Quality\} \\ Attr &= \{color\ depth,\ frame\ rate,\ sampling\ rate, \\ &\quad\ sample\ bits\} \\ Val &= \{\{1, integer, discrete\}, \{3, integer, discrete\}, \\ &\quad \ldots, \{[1, \ldots, 30], integer, continuous\}, \ldots\} \end{aligned}$$

$$\begin{aligned} DA_{Video\ Quality} &= \{color\ depth,\ frame\ rate\} \\ DA_{Audio\ Quality} &= \{sampling\ rate,\ sample\ bits\} \end{aligned}$$

$$\begin{aligned} AV_{color\ depth} &= \{1, 3, 8, 16, 24\} \\ AV_{frame\ rate} &= \{[1, \ldots, 30]\} \\ AV_{sampling\ rate} &= \{8, 16, 24, 44\} \\ AV_{sample\ bits} &= \{8, 16, 24\} \end{aligned}$$

3.2. Service request

It is clearly infeasible to make the user specify the utility of every quality choice, for all the QoS dimensions of a particular application. There are simply too many choices. Instead, we impose a preference order over the dimensions, its attributes and their values on user's service request. While we want a user to provide a semantically rich request, so that the system tries to achieve a service the more closely related to user's preferences, we also want to ensure that a user is actually able to express his preferences in his request.

Suppose that, in a remote surveillance system, video is much more important to the user than audio. Assuming that for a particular user a gray scale, low frame rate is fine for video, his request could be as follows:

1. Video Quality
   (a) frame rate: [10,...,5], [4,...1]
   (b) color depth: 3, 1
2. Audio Quality
   (a) sampling rate: 8
   (b) sample bits: 8

The relative decreasing order of importance imposed in dimensions, attributes and values expresses user's preferences, that is, elements identified by lower indexes are more important than elements identified by higher indexes.

In the example above, video is more important than audio, and frame rate is more important than color depth in the Video Quality dimension. In a similar way, the audio sampling rate is more important than the sampling size. For each of these attributes, a preference order for the accepted values is as well expressed.

4. Coalitions

Coalition formation is a very active research field in multi-agent systems [19][18][5]. Most of this research assumes that agents are fully rational and consider an optimal coalition. However, most systems in the real world are real-time, dynamic and resource bounded. A single node has no complete information about the environment and other nodes. As such it cannot make fully rational reasoning when planning for a coalition. Due to the dynamic nature and time constraints of our scenario a carefully rationalized coalition planning may be useless or less useful by the time the coalition is formed.

A coalition's life cycle can be decomposed in three phases:

Formation: Selection of individual partners, which will compose the coalition, based in its specific skills, resources, costs and availability.

Operation: Control and monitoring of partners' execution, resolution of conflicts and, possibly, the coalition reconfiguration due to partial failures.

Dissolution: Termination of the coalition.

Our work is currently focused in the formation phase and our proposal concerns the development of a framework that supports the automatic coalition formation process.

QoS-aware applications are usually structured in such a way that they can provide different quality levels, which have associated estimations of the needed resources. As such, they can dynamically change the executing quality level. A QoS manager can negotiate with applications the level they have to provide. This negotiation can be based on a contract model, trading of quality by resources. The final goal is to use resources in an efficient way and to maximize system quality, by a cooperative execution of services.

In order to cope with limited resources, an effective resource allocation algorithm is required. The increasing demand on resources and performance of multimedia applications makes it appropriate for wireless nodes to cooperate with each other, and with wired neighbors, to meet resource allocation requests and handle stringent bandwidth constraints.

4.1. Cooperative service allocation

Consider a network with several nodes, each one with its own particular resources R_i. There will be several services S to be executed, each one with a set (for now) of independent tasks T. Each service has specific QoS constraints, defined by the user, and will compete for the finite set of resources R.

Let Q_i be the set of QoS constraints associated with task T_i. Each Q_{kj} is a finite set of quality choices for the j^{th} attribute of dimension k. This can be either a discrete or continuous set.

Each node may supply a distinct set of resources, with different capacities. The nodes in this heterogeneous network will cooperate to achieve a common goal.

The objective of each coalition is to fulfill the resource allocation requests from users. Every member of the coalition provides some of the requested resources, according to its own operation constraints. We view the distributed resource allocation problem as a cooperative process among nodes.

Let $N = 1, \ldots, n$ denote the set of possible nodes available for the application to request resources from. Let P denote the set of user's preferences on the several QoS dimensions for an application, with its independent tasks T. The basic coalition problem can be described as:

Given a set of nodes N and a resource allocation demand enforced by P they have to satisfy, if the resource demand cannot be satisfied by a single node or when a single node handles the request inefficiently, it is necessary for the nodes in the network to cooperate to fulfill the resource demand. With cooperation between nodes, by forming coalitions among themselves, resources can be allocated by splitting application's tasks by a subset of N.

4.2. Coalition formation

The coalition formation process should enable the selection of individual nodes that, based on their own resources and availability, will constitute the best group to satisfy user's QoS requirements. In such a scenario, the adopted au-

tomatic negotiation mechanism has to be powerful enough to satisfy two important requirements:

- Ability to select the most promising nodes that should belong to the coalition. This means that nodes have to negotiate over requirements described through multiple attributes, which imply that the negotiation process must be enhanced with the capability to both evaluate and formulate multi-attribute proposals.
- In coalition formation process, each one of the individual nodes will contribute with at least one of its own resources. All these contributions may be, and they usually are, mutually dependent. The negotiation process has to be able to deal with those inter-dependencies, reaching a coherent solution.

When a user requests a service, with its specific QoS preferences, on a particular node, the *QoS Provider* starts and guides all the negotiation process, playing the role of *Negotiation Organizer*. Those nodes who are willing to belong to the future coalition (may include the node that starts the negotiation) have to submit their multi-attribute proposals, for each service's task.

Our negotiation algorithm is described here:

1. The *Negotiation Organizer* broadcasts the description of each service, as well as user's preferences on each QoS dimension.
2. The *QoS Provider* of each node contacts the *Resource Managers* and reply with a multi-attribute proposal.
3. The *Negotiation Organizer*, using a multi-attribute function, evaluates all received proposals and selects the one that offers the best utility.
4. Relevant data for task execution is sent to winning node(s).

The coalition is formed based on the set of proposals that presents:

- Lowest evaluation value, since it is the solution that includes values closer to the preferred ones. As our objective is to maximize user's perceived utility, each task should be executed by the node that offers the QoS level closer to user's preferences.
- Lowest communication cost.
- Lowest number of distinct nodes in coalition. Coalition operation's complexity increases with the number of distinct members.

5. Proposals formulation

Requests for task execution may arrive dynamically. To guarantee the request locally, the *QoS Provider* executes a local QoS optimization heuristic and formulates its proposal.

All entities that participate in the coalition formation process must provide sufficient resources to try to fulfill user's QoS requirements. Therefore, each individual *QoS Provider* must map QoS constraints to resource requirements, and then reserve resources accordingly (resource reservations are made through the *Resource Managers*). This mapping is inherently difficult. To address this problem we (for now) assume that applications make a reasonable accurate analysis of their resource requirements, made a priori through resource monitoring tools, followed by runtime adaptation.

In order to make a proposal the *QoS Provider* contacts the required Resource Managers for resource availability, using the following algorithm, inspired in the local QoS optimization heuristic of [2]. Let each task T_i have an associated set of preferences, specified by the user in relative decreasing order of preference. For each k QoS dimensions there are n possible attributes, Q_{kn}.

1. Start by selecting user's preferred values for all QoS dimensions
2. While the set of tasks is not schedulable
   (a) For each task T_i receiving service at level $Q_{kj} < Q_{kn}$
   (b) Determine the decrease in local reward resulting from degrading attribute j to $j+1$
   (c) Find task T_m whose decrease is minimum and degrade it to the $j+1$'s level

The local reward is calculated by:

$$r = \begin{cases} n & \text{if task is being served at } Q_{k1} \text{ for all dimensions} \\ n - \sum_{j=1}^{n} penalty_j & \text{if } Q_{kj} > Q_{k1} \end{cases} \quad (1)$$

In equation 1 *penalty* is a parameter that decreases the reward value. This parameter can be defined according to user's own criteria and its value increases with the distance for user's preferred value.

6. Proposals evaluation

The evaluation of received proposals implies taking into consideration multiple attributes, across several QoS dimensions. Attaching utility values to different attributes solves the problem of multi-attribute evaluation. Generally, an evaluation formula is a linear combination of attributes' values, weighted by their corresponding utility values. As such, a multi-attribute evaluation is simply converted in a

single attribute evaluation, where the result of the evaluation function can be seen as this single issue.

However, it can be very difficult to the user to specify absolute numeric values to quantify all the QoS dimension's attributes. A more natural, and realistic way, is to simply impose a preference order over the dimensions, its attributes and their values. Therefore, in this work we consider a proposal to be formulated through the relative importance ($k = 1 \ldots n$) of a set of QoS dimensions. Furthermore, for each dimension a relative importance order of attributes is also specified ($i = 1 \ldots attr_k$), where k is the number of attributes of dimension k. Note that k, i are not the identifiers of dimensions and attributes, but their relative position in user's service request. Identifiers are defined in the application's QoS requirements representation (see section 3.1).

All admissible proposals are evaluated according to equation 2. A proposal is admissible if it can satisfy all the QoS dimensions requested by the user. Dimensions and their attributes are evaluated by decreasing order of importance to the user, that is, dimensions and attributes identified by lower indexes are more important than dimensions and attributes identified by higher indexes.

$$distance = \sum_{k=1}^{n} w_k * dist(Q_k) \tag{2}$$

where n is the number of QoS dimensions and $0 \leq w_k \leq 1$ is the relative importance of QoS dimension k, Q_k, to the user, and can defined as

$$w_k = \frac{n - k + 1}{n} \tag{3}$$

In the equation above, QoS dimensions are presented in relative decreasing order of importance to the user. This order is specified in user's service request, encoding user's preferences in a qualitative way.

For each k dimension's evaluation $dist_k$, we propose an weighted sum of the differences between user's preferred values and the values proposed by a specific node to that dimension's attributes.

$$dist(Q_k) = \sum_{i=1}^{attr_k} w_i * dif(Prop_{ki}, Pref_{ki}) \tag{4}$$

where $attr_k$ is the number of attributes in dimension k.

In equation 4, the function $dif(Prop_{ki}, Pref_{ki})$ quantifies, for an attribute i, the degree of acceptability of the proposed value $Prop_{ki}$, when compared to user's preferred value $Pref_{ki}$ and is defined as

$$dif = \begin{cases} \dfrac{Prop_{ki} - Pref_{ki}}{max(Q_k) - min(Q_k)} \text{ if continuous } Q_{ki} \\ \\ \dfrac{pos(Prop_{ki}) - pos(Pref_{ki})}{length(Q_k) - 1} \text{ if discrete } Q_{ki} \end{cases} \tag{5}$$

If attribute i has a continuous domain, this quantification is a normalized difference between the proposed value and the preferred one.

For discrete domains equation 5 considers the preferences attached to $Prop_{ki}$ and $Pref_{ki}$ by using their relative position in the application's QoS requirements specification. In [13] the authors use the notion of *Quality Index*, defining a bijective function that maps the elements of a discrete domain into integer values. We use a similar approach, by mapping the position (index) of that attribute in the domain specification into $Prop_{ki}$'s and $Pref_{ki}$'s scoring values.

The best proposal is the one that presents the lowest evaluation, since it is the one that contains the attributes' values more closely related to user's preferences, in all QoS dimensions.

7. Conclusions

In this paper we addressed the specification of a generic model for enabling distributed service allocation, without a central authority distributing services among nodes, forming temporary coalitions, considering QoS-aware applications.

Wireless nodes may need to cooperate with neighbor nodes in order to fulfill certain services. Given a set of services to be satisfied, we consider situations where a service is assigned to a group of nodes. Service allocation to several nodes is necessary when the processing cannot be performed by a single node or when a single node performs it inefficiently. For example, playing downloaded movies may require decompression. On the other hand, transmitting data to the Internet from the mobile devices may require compression. Where to perform processing is a quite complex problem. By default, the responsibility associated with data processing is on the mobile device. However, such a default action may suffer time penalty and, possibly, battery energy loss. In the examples above, processing on the server may require additional data communication. Its possible to partition the entire process into tasks and divide them among different devices with spare resources.

Various groups of nodes may have different degrees of efficiency in service execution performance due to different capabilities of their members. As such, service allocation should be done with respect to those differences.

Our coalition formation algorithm selects the nodes that offer the solution that includes values closer to user's multidimensional QoS requirements. Those requirements are described through a semantically rich QoS specification interface for multidimensional QoS provisioning, allowing the user, and applications, to define fine-grained service requests.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments and suggestions. This work was partially supported by FCT (FCT/UI/608).

References

[1] T. F. Abdelzaher, E. M. Atkins, and K. G. Shin. Fault-tolerant broadcasts and related problems. In S. Mullender, editor, *Distributed Systems*. Addison-Wesley, 1993.

[2] T. F. Abdelzaher, E. M. Atkins, and K. G. Shin. Qos negotiation in real-time systems and its application to automated flight control. *IEEE Transactions on Computers, Best of RTAS '97 Special Issue*, 49(11):1170–1183, November 2000.

[3] ARTIST (IST-2001-34820). *Selected topics in Embedded Systems Design: Roadmaps for Research. Part III Adaptive Real-Time Systems for Quality of Service Management*, May 2004. Available at http://www.artist-embedded.org/.

[4] Christina Aurrecoechea, Andrew T. Campbell, and Linda Hauw. A survey of qos architectures. *Multimedia Systems*, 6(3):138–151, 1998.

[5] Silvia Breban and Julita Vassileva. A coalition formation mechanism based on inter-agent trust relationships. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 306–307. ACM Press, 2002.

[6] A. Campbell, G. Coulson, and D. Hutchison. A quality of service architecture. *Computer Communication Review*, 24(2):6–27, April 1994.

[7] Michael Ditze, Berta Batista, Eduardo Tovar, Peter Altenbernd, and Filipe Pacheco. Workload balancing in distributed virtual reality environments. In *1st Intl. Workshop on Real-Time LANs in the Internet Age, Satellite Event to the 14th Euromicro Conference on Real- Time Systems*, 2002.

[8] P. Havinga, G. Smit, L.Vognild, and G. Wu. The smart project: Exploiting the heterogenous mobile world. In *Proceedings of the 2nd International Conference on Internet Computing*, pages 346–352, 2001.

[9] J. M. Hyman, A. A. Lazar, and G. Pacifici. Real-time scheduling with quality of service constraint. *IEEE Journal on Selected Areas in Communications*, 9(7), September 1991.

[10] R. Jain and U. Varshney. Suporting quality of service in multiple heterogeneous wireless networks. In *Proceedings of the IEEE IVehicular Technology Conference*, pages 952–956, 2002.

[11] K. Kawachiya and H. Tokuda. A negotiation-based resource management framework for dynamic qos control. In *Proceedings of Real-Time Mach Workshop*, 1997.

[12] Ulrich Kermer, Jamey Hicks, and James Rehg. A compilation framework for power and energy management on mobile computers. In *14th International Workshop on Parallel Computing*, pages 115–131, 2001.

[13] Chen Lee, John P. Lehoczky, Daniel P. Siewiorek, Ragunathan Rajkumar, and Jeffery P. Hansen. A scalable solution to the multi-resource qos problem. In *IEEE Real-Time Systems Symposium*, pages 315–326, 1999.

[14] Ming Li, B. Prabhakaran, and S. Sathyamurthy. On flow reservation and admission control for distributed scheduling strategies in ieee802.11 wireless lan. In *Proceedings of the 6th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*, pages 108–155, 2003.

[15] Zhiyuan Li, Cheng Wang, and Rong Xu. Computation offloading to save energy on handheld devices: a partition scheme. In *Proceedings of the 2001 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, pages 238–246. ACM Press, 2001.

[16] Mazliza Othman and Stephen Hailes. Power conservation strategy for mobile computers using load sharing. *SIGMOBILE Mobile Computing Communications Review*, 2(1):44–51, 1998.

[17] Alexey Rudenko, Peter Reiher, Gerald J. Popek, and Geoffrey H. Kuenning. Saving portable computer battery power through remote process execution. *Mobile Computing and Communications Review*, 2(1):19–26, 1998.

[18] Tuomas Sandholm and Victor R. Lesser. Coalitions among computationally bounded agents. *Artificial Intelligence*, 94(1-2):99–137, 1997.

[19] Onn Shehory and Sarit Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200, 1998.

[20] P. Steenkiste, A. Fisher, and H. Zhang. Resource management in application-aware networks. *Workshop on the Integration of IP and ATMM*, 1997.

[21] C. Volg, L. Wolf, R. Herrtwich, and H. Wittig. Heirat - quality of service management for distributed multimedia systems. *Multimedia Systems Journal*, 1995.

[22] Clemens C. Wüst, Liesbeth Steffens, Reinder J. Bril, and Wim F.J. Verhaegh. Qos control strategies for high- quality video processing. In *Proceedings of the 16th Euromicro Conference on Real-Time Systems*, Catany, Italy, June 2004.

[23] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. Rsvp: A new resource reservation protocol. *IEEE Network*, pages 8–18, September 1993.

[24] Hua Zhu, Ming Li, Imrich Chlamtac, and B. Prabhakaran. A survey of quality of service in ieee 802.11 networks. *Wireless Communications, IEEE*, 11(4):6–14, 2004.