J Supercomput (2011) 55: 367–399
DOI 10.1007/s11227-009-0330-5

Tradeoffs between energy consumption and QoS in mobile grid

Chunlin Li · Layuan Li

Published online: 10 October 2009

Abstract Mobile grid, which combines grid and mobile computing, supports mobile users and resources in a seamless and transparent way. However, mobility, QoS support, energy management, and service provisioning pose challenges to mobile grid. The paper presents a tradeoff policy between energy consumption and QoS in the mobile grid environment. Utility function is used to specify each QoS dimension; we formulate the problem of energy and QoS tradeoff by utility optimization. The work is different from the classical energy aware scheduling, which usually takes the consumed energy as the constraints; our utility model regards consumed energy as one of the components of measure of the utility values, which indicates the tradeoff of application satisfaction and consumed energy. It is a more accurate utility model for abstracting the energy characteristics and QoS requirement for mobile users and resources in mobile grid. The paper also proposes a distributed energy–QoS tradeoff algorithm. The performance evaluation of our energy–QoS tradeoff algorithm is evaluated and compared with other energy and deadline constrained scheduling algorithm.

Keywords Grid · QoS · Tradeoff · Mobile computing

1 Introduction

The current grid architecture and algorithms do not take into account the mobile computing environment since mobile devices have not been seriously considered as valid

C. Li (✉) · L. Li
Department of Computer Science, Wuhan University of Technology, Wuhan 430063, People's Republic of China
e-mail: chunlin74@tom.com

L. Li
e-mail: jwtu@public.wh.hb.cn

computing resources or interfaces in grid communities. Just recently, attention has been given to integrate these two emerging techniques of mobile and grid computing [1–9]. Mobile grid, which combines grid and mobile computing, supports mobile users and resources in a seamless, transparent, secure, and efficient way. The interest to incorporate mobile devices into grid systems has arisen with two main purposes. The first one is to enrich users of these devices while the other is that of enriching the own grid infrastructure. Grid offers its services to the mobile users to complete their works in a fast and simple way and on the other hand, the mobile devices offer their limited resources in any place and at any time that is endorsed by the fast advance in the yield and capacity that is being carried out in the mobile technology. Even though mobile devices have limited resources compared to their stationary counterparts, they seem to increasingly gain sufficiently powerful CPUs and storage means. In effect, they are considered capable of providing useful resources. Since the number of mobile devices continuously increases, the aggregate of their resources cannot be considered negligible. Although mobile devices promote mobile communication and flexible use, they still bring problems such as unpredictable quality of the network, low confidence, limited resources (energy, bandwidth, etc.), and periods of disconnections. Clearly, many issues become the challenges when we consider mobile devices as one of grid computing resources or grid users. Some examples of limitations that possibly hinder the integration are relatively poor local resources (in terms of computation speed, memory), battery constraints, unreliable connectivity status, weak security, and so on. These limitations and constraints should be dealt with accordingly before mobile grid integration is fully enabled.

Qualities of service (QoS) are often associated with jobs submitted to a mobile grid system. An example of this is jobs with user-defined deadlines. The QoS of a job is satisfied if it finishes before the specified deadline. In order to make mobile grid reality, mobility, QoS support, energy management, and service provisioning pose challenges. There is an inherent conflict in the design goals for high grid application QoS and low energy consumption. For example, multimedia applications have distinctive quality of service (QoS) and processing requirements which tend to make them extremely resource-hungry and energy intensive. Consequently, delivering high quality multimedia content to mobile handheld devices while preserving their energy presents competing design requirements. Energy saving and low latency are two conflict objectives; a processing node that provides a task with the earliest finish time may not be an ideal candidate in terms of energy saving. This is because the execution time of a task is inverse proportion to the processing speed of a node while the energy consumption is direct proportion to processing speed. Some schemes aim at getting the shortest schedule length that may waste a lot of energy. Other schemes prefer to use low energy consumption sensors that are sufficient to just meet their deadline.

The paper presents a tradeoff policy between energy consumption and QoS in a mobile grid environment. Utility function is used to specify each QoS dimension; we model tradeoff policy between energy consumption and QoS by utility optimization. The work is different from the classical energy aware scheduling, which usually takes the consumed energy as constraints. Our utility model regards consumed energy as one of the components of measure of the utility values, which indicates the tradeoff of application satisfaction and consumed energy; it is a more accurate utility model for abstracting the energy characteristics for mobile users and resources in mobile grid. The paper proposes a distributed energy–QoS tradeoff algorithm. The performance evaluation of our energy–QoS tradeoff algorithm is evaluated and compared with other energy and deadline constrained scheduling algorithms.

The rest of the paper is structured as follows. Section 2 discusses the related works. Section 3 presents tradeoffs between energy consumption and QoS in a mobile grid. Section 4 presents the energy–QoS tradeoff algorithm. In Sect. 5, the experiments are conducted and discussed. Section 6 discusses the integration into EGEE. Section 7 gives the conclusions to the paper.

2 Related works

Recently, energy aware scheduling has been widely studied. Park et al. [4] propose to exploit a grid infrastructure to extend the battery life of mobile devices based on the contexts of mobile applications, devices, and grid systems. They present a framework, called selective grid access (SGA), to optimally utilize the limited resources of mobile devices and grids. SGA aims to reconfigure the placement of mobile applications on either the mobile device or grid nodes and to adjust the QoS level according to the current context of the residual energy and the resource availability of the grids. In [5], the concept of a decentralized job scheduler has been proposed which supports the integration of unstable mobile devices as computational grid resources. The concept has been applied to the demanding scenario of mobile ad hoc grids. Scheduling decisions are delegated to the self-monitoring mobile worker peers that decide based on rules specified by policies. In [6], Wong and Ng presented the performance evaluation of Mobile Grid Services developed by using the MGS application-programming interface. The MGS API, constructed by combining an existing mobile agent system (JADE) and a generic grid system toolkit (Globus), is proposed to support the development of Mobile Grid Services (extended Grid services with mobility during execution). Katsaros and Polyzos [10] investigated the fundamental issues rising in the path toward the realization of the Mobile Grid paradigm. They discussed various approaches in literature and pointed out the problems introduced by node mobility. They studied the performance of a hierarchical, campus-wide networking architecture based on appropriate traces. Huang et al. [3] present techniques for exploiting intermittently available resources in grid infrastructures to support QoS-based multimedia applications on mobile devices. They integrate power aware admission control, grid resource discovery, dynamic load-balancing, and energy adaptation techniques to enable power deficient devices such as to run distributed multimedia applications. Zong and Qin [2] design energy efficient scheduling algorithms for parallel applications running on clusters. They propose a scheduling strategy called the energy efficient task duplication schedule, which can significantly conserve power by judiciously shrinking communication energy cost when allocating parallel tasks to heterogeneous computing nodes. AlEnawy and Aydin [7] propose to minimize the number of dynamic failures while remaining within the energy budget. They propose techniques to statically compute the speed of the CPU in order to meet the

(m, k)-firm deadline constraints. Xie et al. [8] address the issue of allocating tasks of parallel applications in heterogeneous embedded systems with an objective of energy saving and latency reducing. They proposed BEATA (Balanced Energy-Aware Task Allocation), a task allocation scheme considering both energy consumption and schedule length, is developed to solve the energy-latency dilemma. Kim et al. [9] provide power-aware scheduling algorithms for bag of tasks applications with deadline constraints on DVS enabled cluster systems in order to minimize power consumption as well as to meet the deadlines specified by application users. The main differences between other works and our work [11–15] are from three aspects. Firstly, the paper addresses the problem of the energy–QoS tradeoff scheme for the mobile grid environment. We investigate both energy minimization for mobile devices and grid utility optimization problems. Secondly, we solve the tradeoff between energy and the deadline to find a system-wide optimization by using a utility decomposition method. Thirdly, the paper adopts a pricing based iterative algorithm for energy constraint scheduling. The above three contributions do not appear in other related works.

3 Tradeoffs between energy consumption and QoS in mobile grid

3.1 Model description

The paper formulates the energy consumption and QoS tradeoff in the mobile grid by adopting a computational economy framework. The proposed model consists of two types of agents: the grid resource agents that represent the economic interests of the underlying resources providers of the mobile grid and the grid user agents that represent the interests of grid user application using the grid to achieve goals. Interactions between the two agent types are mediated by means of market mechanisms. Market mechanism in economics is based on distributed self-determination, the variation of price reflects the supply and demand of resources, and market theory in economics provides precise depiction for efficiency of resource scheduling. Grid user agents are allowed to specify their requirements and preference parameters by a utility model. In our model, a utility function can be specified for each QoS dimension. We model each of these diverse requirements as a quality of service (QoS) dimension of a job. As a result, a market-based mobile grid model inherently supports grid users with diverse requirements for the execution of their jobs. The utility values are calculated by the supplied utility function that can be formulated with the job parameters. The request is analyzed by the scheduler of the grid market. Whenever a new grid user agent is created, it is first given an endowment of electronic cash to spend to complete its job. A job can be characterized by deadline, budget, data size, and runtime requirements. The budget is the amount of money that the consumer promises to pay for the completion of the job. The grid market mechanism allows multiple grid resource agents and grid user agents to negotiate simultaneously. It uses a price-directed approach to allocate appropriate grid resources. In this price-directed approach, an initial set of prices is announced to the grid user agent. Grid users could update their allocations based on the resource provider's price policy, and iteratively approach an optimal solution. In each iteration, grid user agents individually determine their optimal

Table 1 The description of notations

Notations	Meanings
y_i^k	network allocation obtained by grid application i from the network resource k
x_i^j	computing power obtained by grid application i from computing power j
e_i^l	energy obtained by grid application i from energy resource l
Ce_l	capacity of energy resource l
Cn_k	capacity of network resource k
Cc_j	capacity of computing power j
t_i^n	the time taken by the grid application i to complete nth job
Pc_i^j	the payments of the grid application i to the computing power j
Pn_i^k	the payments of the grid application i to the network resource k
Pe_i^l	the payments of the grid application i to energy resource l
B_i	the expense budget of grid application i
er_l	the energy consumption rate of energy resource l
e_i^n	energy dissipation caused by grid application i's nth job
cq_i^n	computation task of ith grid application's nth job
bq_i^n	transmission task of ith grid application's nth job
eq_i^n	energy storage task of ith grid application's nth job
np_k	the price of network resource k
cp_j	the price of computing resource j
ep_l	the price of energy resource l
T_i	deadlines given by the grid application i to complete its all jobs

allocation and communicate their results to the grid resource agents. Grid resource agents then update their prices and communicate the new prices to the user agents and the cycle repeats. Prices are then iteratively changed to accommodate the demands for resources until the total demand equals the total amount of resources available. We assume that when a grid user agent purchases a portion of resources owned by the resource agents, it is guaranteed that the user agent continues to receive the resource uninterrupted from the resource agent until its task is completed. Grid resource agents publish resource descriptions to the grid market. Resource providers compete actively for jobs from resource consumers and execute them for gaining profits. Every provider tries to maximize its profit based on its resource capability. We assume that the grid resource agents do not cooperate. Instead, they act noncooperatively with the objective of maximizing their individual profits. The grid resource agents compete among each other to serve the grid user agents. The grid user agents do not collaborate either, and try to purchase as much grid resources as possible with the objective of maximizing their net benefit.

3.2 Mathematic formulation for energy consumption and QoS tradeoffs

The notations used in the following sections are listed in Table 1.

In the mobile grid environment, QoS of all applications running in mobile devices should be controlled, so that they do not exhaust the resources of the device on the mobile grid, including residual battery energy, memory, and CPU capacity. It is important for the mobile grid system to manage energy consumption without compromising the system's performance. The paper considers tradeoffs between energy consumption and QoS in the mobile grid environment.

It is assumed that the mobile grid system consists of multiple grid sites that contain mobile nodes and ordinary fixed grid nodes. Mobile nodes consist of a collection of mobile devices M connected by a wireless network. The set M contains n mobile devices, labeled as $m_1, m_2, \ldots, m_n$. Each site may contain multiple mobile nodes and computing nodes. The mobile devices in the grid system may have different resources such as network, computing power, and energy. A mobile device m has an application set $A = \{A_1, A_2, \ldots, A_i\}$ and a resource set $R = \{R_1, R_2, R_n, \ldots\}$. C_i is the available capacity of the resource R_i. The relation between QoS and the resource consumption can be utilized to set dynamic QoS parameters. A mobile device estimates its energy consumption rate er_l for executing the application set $A = \{A_1, A_2, \ldots, A_i\}$, and the energy consumption constraint is C_l. For instance, the energy limitation of a mobile device imposes a constraint as follows:

$$er_l t_i \le C_l$$

where t_i is the completion time of application i.

The processing power of a mobile device m_i is measured by the average CPU speed. For any mobile device $m_i \in M$, there are grid jobs arriving at m_i. The jobs are assumed to be computationally intensive, mutually independent, and can be executed at any mobile device. As soon as a job arrives, it must be assigned to one mobile device for processing. When a job is completed, the executing mobile device will return the results to the originating mobile device or ordinary fixed grid node of the job. We use J to denote the set of all jobs generated by grid application i, $J_i = \{J_i^1, J_i^2, \ldots, J_i^n\}$. Each grid job can be described as $J_i^n = (t_i^n, e_i^n)$, in which t_i^n stands for the time taken by the ith grid application to complete nth job and e_i^n stands for energy dissipation of nth job. There are no dependencies among the jobs, so the submission order and completion order will not impact on the execution result. A user application set is represented as $A = \{A_1, A_2, \ldots, A_i\}$, for $1 \le i \le N$, grid application A_i submits a job, together with parameters including: T_i, which is the deadline limit of job completion time, B_i, which is the expense budget limit for all jobs, and E_i, which is a limited energy budget for all jobs.

The energy consumption rate of each node in the system is measured by Joule per unit time. Let e_i^n be an energy dissipation caused by grid application i's nth job and t_i^n is the execution time of job n of grid application i on the grid node. $er$is the energy consumption rate of energy resource l. If the energy consumption is proportional to execution time of job n, as is the case with battery energy, the energy dissipation of grid application i's nth job can be written as

$$e_i^n = ert_i^n \tag{3.1}$$

We assume that the mobile grid has heterogeneous nodes with different system performance rates and network conditions. This means that the energy consumption of the mobile device can vary with the response time of the application and the network bandwidth. We denote e_i^l is the consumed energy fraction of the energy l (e.g., a battery) by grid application i. Total consumed energy of all grid applications $\sum_{i=1}^{I} e_i^l$ does not exceed the total capacity Ce_l of energy l. Thus, the following resource consumption constraint needs to be satisfied:

$$\sum_{i=1}^{I} e_i^l \le Ce_l \tag{3.2}$$

We define the energy consumption of each application A_i as the sum of the energy consumed by N grid jobs $\sum_{n=1}^{N} e_i^n$. The energy consumption of all grid jobs of each application A_i should be less than the available resources of e_i^l which is a limited energy budget of grid user application i. For each grid application A_i, the consumed energy of all grid jobs of A_i should satisfy

$$\sum_{n=1}^{N} e_i^n \le e_i^l \tag{3.3}$$

The utility concept from microeconomics and game theory can be used to resource scheduling. Utility is a measure of a customer's degree of satisfaction and can be modeled as a function of the quality of service he has received and as well as money he has paid. A utility function is used for users to specify their preferences. The utility function can be used to deal with the heterogeneity and load variations of grid environment. The choice of utility function depends on the user application (e.g., data, video). Now, we formulate the problem of tradeoffs between energy consumption and QoS in mobile grid as the constraint optimization problem; the utility of the system U_{system} is defined as the sum of grid application utilities.

$$U_{\text{system}} = \sum_{i=1}^{I} U_i\left(e_i^l, x_i^j, y_i^k\right) \tag{3.4}$$

Where e_i^l is the energy obtained by grid application i from the energy l. x_i^j is CPU allocation obtained by grid application i from the computing resource provider j. y_i^k is bandwidth allocation obtained by grid application i from the network resource provider k. The utility function for application A_i depends on allocated resources x_i^j, y_i^k, and consumed energy e_i^l. The objective of energy–QoS tradeoff scheduling is to maximize the utility of the system U_{system} without exceeding the resource capacity, the energy budget, expense budget, and the deadline. We formalize the problem using a nonlinear optimization theory; the energy–QoS tradeoff in mobile grid can be formulated as follows:

$$\text{Max} \quad U_{\text{system}} = \sum_{i=1}^{I} U_i\left(e_i^l, x_i^j, y_i^k\right)$$

$$\text{s.t}\quad B_i \geq \sum_{l=1}^{L} Pe_i^l + \sum_{j=1}^{J} Pc_i^j + \sum_{k=1}^{K} Pn_i^k, \qquad \sum_{i=1}^{I} e_i^l \leq Ce_l$$

$$T_i \geq \sum_{n=1}^{N} t_i^n, \qquad Cn_k \geq \sum_{i=1}^{I} y_i^k, \qquad \sum_{n=1}^{N} e_i^n \leq e_i^l, \qquad Cc_j \geq \sum_{i=1}^{I} x_i^j \tag{3.5}$$

In (3.5), the first type of constraints is related with a different resource capacity. The QoS constraint implies that the aggregate network resource units $\sum_{i=1}^{I} y_i^k$ do not exceed the total capacity Cn_k of the network resource provider k, aggregate consumed energy of all grid application $\sum_{i=1}^{I} e_i^l$ does not exceed the total Ce_l of energy l, and aggregate computing power $\sum_{i=1}^{I} x_i^j$ does not exceed the total resource Cc_j of the computing resource provider j. The second type of constraints is related with the grid application expense budget. Grid application needs to complete a sequence of jobs in a specified amount of time, T_i, while the payment overhead accrued cannot exceed B_i, which is the expense budget of grid application i. Pe_i^l, Pc_i^j, Pn_i^k are the payments of the grid application i to the energy storage provider l, computing resource provider j and network resource provider k. The total payments of the grid application i $\sum_{l=1}^{L} Pe_i^l + \sum_{j=1}^{J} Pc_i^j + \sum_{k=1}^{K} Pn_i^k$ does not exceed B_i. The total energy consumed by all jobs of grid application i $\sum_{n=1}^{N} e_i^n$ cannot exceed the energy budget e_i^l which is the available energy obtained by grid application i from the energy storage l.

We can apply the Lagrangian method to solve such a problem (3.5). The Lagrangian approach is used to solve constrained optimization problems. Luh and Hoitomt [16] successfully adopted the Lagrangian approach by breaking the overall manufacturing problem into a series of subproblems. This approach combines Lagrangian relaxation techniques (Appendix) with scheduling heuristics. Let us consider the Lagrangian form of energy–QoS tradeoff optimization problem:

$$\begin{aligned} L\left(e_i^l, x_i^j, y_i^k\right) = {} & \sum_{i=1}^{I} U_i\left(e_i^l, x_i^j, y_i^k\right) - \lambda_i \left(\sum_{i=1}^{I} e_i^l - Ce_l\right) - \beta_i \left(\sum_{i=1}^{I} x_i^j - Cc_j\right) \\ & - \varphi_i \left(\sum_{i=1}^{I} y_i^k - Cn_k\right) - \gamma_i \left(\sum_{l=1}^{L} Pe_i^l + \sum_{j=1}^{J} Pc_i^j + \sum_{k=1}^{K} Pn_i^k - B_i\right) \\ & - \mu_i \left(\sum_{n=1}^{N} e_i^n - e_i^l\right) - \alpha_i \left(\sum_{n=1}^{N} t_i^n - T_i\right) \end{aligned} \tag{3.6}$$

where λ_i, β_i, and φ_i are the Lagrange multipliers of grid application with their interpretation of energy price, computing resource capacity price, and network resource capacity price, respectively. Since the Lagrangian is separable, this maximization of Lagrangian over (x_i^j, y_i^k, e_i^l) can be conducted in parallel at each application A_i. In (3.5), though the allocated resources x_i^j, y_i^k, and consumed energy e_i^l are coupled in their constraints, respectively, they are separable. Given that the grid knows the utility functions U of all the grid applications, this optimization problem can be mathematically tractable. However, in practice, it is not likely to know each application's utility, and it is also infeasible for the mobile grid environment to compute and allocate resources in a centralized fashion. In order to derive a distributed algorithm to solve (3.5), we decompose the problem into the subproblems.

In this paper, maximization formulation of the grid system utility adopts a network utility maximization (NUM) framework [17] in which each application has an associated utility function. In [17], an optimization framework leads to a decomposition of the overall system problem into a separate problem for each user, in which the user chooses a charge per unit time that the user is willing to pay, and one for the network. The network's optimization problem leads to two classes of the algorithm, which may be interpreted in terms of either congestion indication feedback signals or explicit rates based on shadow prices. It was shown that a system optimum is achieved when users' choices of charges and the network's choice of allocated rates are in equilibrium.

The grid system utility denoted as the sum of grid application utility can be defined as follows (3.7):

$$\begin{aligned} U_{\text{system}} &= \sum_{i=1}^{I} U_i\left(e_i^l, x_i^j, y_i^k\right) \\ &= \left(B_i - \sum_{l=1}^{L} Pe_i^l - \sum_{j=1}^{J} Pc_i^j - \sum_{k=1}^{K} Pn_i^k\right) + \left(T_i - \sum_{n=1}^{N} t_i^n\right) \\ &\quad + \sum_{i=1}^{I}\left(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k\right) + \left(e_i^l - \sum_{n=1}^{N} e_i^n\right) \end{aligned} \tag{3.7}$$

Grid system utility functions are maximally optimized with specific constraints. In (3.7), $Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k$ present the revenue of the energy storage resource, computing power, and network resource provider. We could have chosen any other form for the utility that increases with x_i^j, y_i^k, e_i^l. But we chose the log function because the benefit increases quickly from zero as the total allocated resource increases from zero and then increases slowly. Moreover, log function is analytically convenient, increasing, strictly concave, and continuously differentiable. The benefits of a grid resource provider are affected by payments of grid applications and allocated resources. It means that the revenue increases with increasing allocated resources and increasing payment.

The Lagrangian form of (3.5) can be reformulated as follows (3.8):

$$\begin{aligned} L\left(e_i^l, x_i^j, y_i^k\right) &= \left(B_i - \sum_{l=1}^{L} Pe_i^l - \sum_{j=1}^{J} Pc_i^j - \sum_{k=1}^{K} Pn_i^k\right) + \left(T_i - \sum_{n=1}^{N} t_i^n\right) \\ &\quad + \left(e_i^l - \sum_{n=1}^{N} e_i^n\right) + \sum_{i=1}^{I}\left(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k\right) \end{aligned}$$

$$\begin{aligned}
&-\lambda_i\left(\sum_{i=1}^{I} e_i^l - Ce_l\right) - \beta_i\left(\sum_{i=1}^{I} x_i^j - Cc_j\right) \\
&-\gamma_i\left(\sum_{l=1}^{L} Pe_i^l + \sum_{j=1}^{J} Pc_i^j + \sum_{k=1}^{K} Pn_i^k - B_i\right) - \mu_i\left(\sum_{n=1}^{N} e_i^n - e_i^l\right) \\
&-\alpha_i\left(\sum_{n=1}^{N} t_i^n - T_i\right) - \varphi_i\left(\sum_{i=1}^{I} y_i^k - Cn_k\right) \qquad (3.8)
\end{aligned}$$

The system model presented by (3.5) is a nonlinear optimization problem with N decision variables. Since the Lagrangian is separable, the maximization of the Lagrangian can be processed in parallel for grid user applications and grid resource providers, respectively. From (3.8), the resource allocation $\{e_i^l, x_i^j, y_i^k\}$ solves (3.5) if and only if there exist a set of nonnegative shadow costs $\{\lambda_i, \beta_i, \varphi_i\}$. Generally solving such a problem by typical algorithms such as steepest decent method and gradient projection method is of high computational complexity, which is very time costing and impractical for implementation. In order to reduce the computational complexity, we decompose the utility optimization problem (3.5) into two subproblems for grid user applications and grid resource providers so that the computational complexity is reduced. The shadow costs suggest a mechanism to distribute the resource optimization between the grid applications and the grid system. The problem (3.5) maximizes the utility of grid applications on the energy price, computing power capacity price, and network resource capacity price, $\sum_{i=1}^{I} U_i(e_i^l, x_i^j, y_i^k)$ is the total utility of mobile grid system, $\beta_i \sum_{i=1}^{I} x_i^j$ is the computing power cost, $\lambda_i \sum_{i=1}^{I} e_i^l$ is the energy cost, and $\varphi_i \sum_{i=1}^{I} y_i^k$ is the network resource cost. By decomposing the Kuhn–Tucker conditions into separate roles of consumer and supplier at grid market, the centralized problem (3.5) can be transformed into a distributed problem. Grid application's payment is collected by the resource providers. The payments of grid applications paid to resource providers are the payments to resolve the optimality of resource allocation in the grid market. We decompose the problem into the following two subproblems (3.9) which is the grid user application QoS optimization problem and (3.10) which is the grid resource providers' optimization problem, and seek a distributed solution where the grid resource provider does not need to know the utility functions of individual grid user application. Equations (3.9) and (3.10) derived from the distributed approach are identical to the optimal conditions given by the centralized energy–QoS tradeoff optimization problem (3.5). This means if two subproblems converge to its optimal points, then a globally optimal point is achieved. Total user application benefit of the mobile grid is maximized when the equilibrium prices, obtained through the two subproblems (3.9) and (3.10) equal the Lagrangian multipliers λ_i, β_i, and φ_i, where λ_i, β_i, and φ_i are the optimal prices charged by resource providers including energy, computing power, and network resource to grid applications. Two maximization subproblems correspond to grid user application QoS optimization problem as denoted by (3.9)

$$\text{Max } U_{\text{app}} = \left(B_i - \sum_{l=1}^{L} Pe_i^l - \sum_{j=1}^{J} Pc_i^j - \sum_{k=1}^{K} Pn_i^k\right) + \left(T_i - \sum_{n=1}^{N} t_i^n\right) + \left(e_i^l - \sum_{n=1}^{N} e_i^n\right)$$

$$\text{App:} \quad \sum_{n=1}^{N} e_i^n \le e_i^l, \qquad T_i \ge \sum_{n=1}^{N} t_i^n, \qquad B_i \ge \sum_{l=1}^{L} Pe_i^l + \sum_{j=1}^{J} Pc_i^j + \sum_{k=1}^{K} Pn_i^k \tag{3.9}$$

$$\begin{aligned} &\text{Resource:} \quad \text{Max } U_{\text{resource}} = \sum_{i=1}^{I} \left(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k\right) \\ &\qquad \sum_{i=1}^{I} e_i^l \le Ce_l, \qquad Cc_j \ge \sum_{i=1}^{I} x_i^j, \qquad Cn_k \ge \sum_{i=1}^{I} y_i^k \end{aligned} \tag{3.10}$$

In problem (3.9), the grid application gives the unique optimal payment to the resource provider under the energy budget, expense budget, and the deadline constraint to maximize the user's satisfaction. $(B_i - \sum_{l=1}^{L} Pe_i^l - \sum_{j=1}^{J} Pc_i^j - \sum_{k=1}^{K} Pn_i^k)$ represents the money surplus of grid application i, which is obtained by expense budgets subtracting the payments to various resource providers. $(T_i - \sum_{n=1}^{N} t_i^n)$ represents the saving times for grid application i, which is obtained by time limit subtracting actual spending time. $(e_i^l - \sum_n e_i^n)$ represents the energy surplus of grid application i which is obtained by the energy budgets subtracting actual energy dissipation. So, the objective of problem (3.9) is to get more surpluses of money and more energy savings, at the same time, complete the task for grid user application as soon as possible. In problem (3.10), different resource providers compute optimal resource allocation for maximizing the revenue of their own. Grid application i submits the payment Pe_i^l to the energy resource provider l, Pn_i^k to network resource provider k and Pc_i^j to computing power provider j. $Pe_i^l \log e_i^l$ presents the revenue obtained by energy resource l from grid application i. $Pc_i^j \log x_i^j$ presents the revenue obtained by computing power j from grid application i. $Pn_i^k \log y_i^k$ presents the revenue obtained by network resource k from grid application i. The objective of resource providers is to maximize $Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k$ under the constraints of their provided resource amounts. Grid resource providers cannot sell the resources to grid applications more than the total capacity.

In problem (3.9), the grid application adaptively adjusts its payments to computing power, network resource, and energy based on the current resource conditions, while in problem (3.10), the grid resource provider adaptively allocates energy, CPU, and bandwidth required by the grid application in the problem (3.9). The interaction between two subproblems is controlled through the use of the price variable λ_i, β_i, and φ_i, which is the energy price, computing power price, and network resource price charged from grid applications by the grid energy resource, computing power, and network resource. The interaction between two subproblems also coordinates the grid application's payment and the supply of grid resource providers.

3.3 Energy–QoS tradeoff optimization in mobile grid

In problem (3.9), grid application maximizes its satisfaction and gives the unique optimal payment to the resource provider under the energy budget, expense budget, and the deadline constraint. We assume that grid application i submits Pe_i^l to energy resource l, Pc_i^j to computing power j, and Pn_i^k to network resource k. Then $Pe_i = [Pe_i^1 \ \dots \ Pe_i^l]$ represents all payments of grid applications for energy resource $l, Pc_i = [Pc_i^1 \ \dots \ Pc_i^j]$ represents all payments of grid applications for computing power j, and $Pn_i = [Pn_i^1 \ \dots \ Pn_i^k]$ represents all payments of grid applications for the network resource k. Let $m_i = \sum_l Pe_i^l + \sum_j Pc_i^j + \sum_k Pn_i^k$; m_i is the total payment of the ith grid application. N grid applications compete for grid resources with finite capacity. The resource is allocated using a market mechanism, where the partitions depend on the relative payments sent by the grid applications. Let ep_l, cp_j, np_k denote the price of the resource unit of energy resource l, the price of the resource unit of computing power j, and network resource k, respectively. Let the pricing policy, $ep = (ep_1, ep_2, \dots, ep_l)$, denote the set of resource unit prices of all the energy resources in the grid, $cp = (cp_1, cp_2, \dots, cp_j)$, denote the set of resource unit prices of all the computing powers, $np = (np_1, np_2, \dots, np_k)$ is the set of network resource unit prices. The ith grid application receives the resources proportional to its payment relative to the sum of the resource provider's revenue. Let e_i^l, x_i^j, y_i^k be the fraction of resource units allocated to grid application i by energy l, computing power j, and network resource k.

$$x_i^j = Cc_j \frac{Pc_i^j}{cp_j}, \qquad e_i^l = Ce_l \frac{Pe_i^l}{ep_l}, \qquad y_i^k = Cn_k \frac{Pn_i^k}{np_k}$$

The time taken by the ith grid application to complete nth job is

$$t_i^n = \frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}$$

The energy dissipation used by the ith grid user to complete nth job is

$$e_i^n = ert_i^n = er\left(\frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}\right)$$

Problem (3.9) can be reformulated as

$$\begin{aligned}\text{Max } U_{\text{app}} = {} & \left(B_i - \sum_{l=1}^{L} Pe_i^l - \sum_{j=1}^{J} Pc_i^j - \sum_{k=1}^{K} Pn_i^k\right) \\ & + \left(T_i - \sum_{n=1}^{N}\left(\frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}\right)\right) \\ & + \left(e_i^l - \sum_{n=1}^{N} er\left(\frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}\right)\right)\end{aligned}$$

The Lagrangian for the grid user's utility is $L_1(Pe_i^l, Pc_i^j, Pn_i^k)$.

$$\begin{aligned}
L_1\big(Pe_i^l, Pc_i^j, Pn_i^k\big) = {}& \left(B_i - \sum_{l=1}^{L} Pe_i^l - \sum_{j=1}^{J} Pc_i^j - \sum_{k=1}^{K} Pn_i^k\right) \\
& + \left(T_i - \sum_{n=1}^{N}\left(\frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}\right)\right) \\
& + \left(e_i^l - \sum_{n=1}^{N} er\left(\frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}\right)\right) \\
& + \nu_i\left(B_i - \sum_{l=1}^{L} Pe_i^l - \sum_{j=1}^{J} Pc_i^j - \sum_{k=1}^{K} Pn_i^k\right) \\
& + \sigma_i\left(T_i - \sum_{n=1}^{N}\left(\frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}\right)\right) \\
& + \varepsilon_i\left(e_i^l - \sum_{n=1}^{N} er\left(\frac{cq_i^n cp_j}{Cc_j Pc_i^j} + \frac{bq_i^n np_k}{Cn_k Pn_i^k} + \frac{eq_i^n ep_l}{Ce_l Pe_i^l}\right)\right)
\end{aligned}$$

where $\varepsilon_i, \sigma_i, \nu_i$ is the Lagrangian constant. From the Karush–Kuhn–Tucker theorem, we know that the optimal solution is given:

$$\frac{\partial L_1(Pe_i^l, Pc_i^j, Pn_i^k)}{\partial Pe_i^l} = 0 \quad \text{for } \varepsilon_i, \sigma_i, \nu_i > 0$$

$$\begin{aligned}
\frac{\partial L_1(Pe_i^l, Pc_i^j, Pn_i^k)}{\partial Pe_i^l} = {}& -1 - \nu_i + \frac{eq_i^n ep_l}{Ce_l (Pe_i^l)^2} + er\frac{eq_i^n ep_l}{Ce_l (Pe_i^l)^2} \\
& + \sigma_i \frac{eq_i^n ep_l}{Ce_l (Pe_i^l)^2} + \varepsilon_i er \frac{eq_i^n ep_l}{Ce_l (Pe_i^l)^2}
\end{aligned}$$

Let $\partial L_1(Pe_i^l, Pc_i^j, Pn_i^k)/\partial Pe_i^l = 0$ to obtain

$$Pe_i^l = \left(\frac{(1 + er + \sigma_i + \varepsilon_i er)eq_i^n ep_l}{(1+\nu_i)Ce_l}\right)^{1/2}$$

Using this result in the constraint equation, we can determine $\theta = (1 + er + \sigma_i + \varepsilon_i er)/1 + \nu_i$ as

$$(\theta)^{-1/2} = \frac{T_i}{\sum_{m=1}^{N}\left(\frac{ep_m eq_i^n}{Ce_m}\right)^{1/2}}$$

We obtain $Pe_i^{l^*}$

$$Pe_i^{l^*} = \left(\frac{eq_i^n ep_l}{Ce_l}\right)^{1/2} \frac{\sum_{m=1}^N \left(\frac{eq_i^n ep_m}{Ce_m}\right)^{1/2}}{T_i}$$

It means that grid application wants to pay $Pe_i^{l^*}$ to energy resource l for needed energy consumed to execute grid jobs under the completion time constraint.

$$\frac{\partial L_1(Pe_i^l, Pc_i^j, Pn_i^k)}{\partial Pc_i^j} = -1 + \frac{cq_i^n cp_j}{Cc_j(Pc_i^j)^2} + er_i^n \frac{cq_i^n cp_j}{Cc_j(Pc_i^j)^2} - \nu_i + \sigma_i \frac{cq_i^n cp_j}{Cc_j(Pc_i^j)^2} + \varepsilon_i er \frac{cq_i^n cp_j}{Cc_j(Pc_i^j)^2}$$

Let $\partial L_1(Pe_i^l, Pc_i^j, Pn_i^k)/\partial Pc_i^j = 0$ to obtain

$$Pc_i^j = \left(\frac{(1 + er + \sigma_i + \varepsilon_i.er)cq_i^n cp_j}{(1 + \nu_i)Cc_j}\right)^{1/2}$$

Using this result in the constraint equation, we can determine $\xi = (1 + er + \sigma_i + \varepsilon_i er)/1 + \nu_i$ as

$$(\xi)^{-1/2} = \frac{T_i}{\sum_{m=1}^N \left(\frac{cp_m cq_i^n}{Cc_m}\right)^{1/2}}$$

We obtain $Pc_i^{j^*}$

$$Pc_i^{j^*} = \left(\frac{cq_i^n cp_j}{Cc_j}\right)^{1/2} \frac{\sum_{m=1}^N \left(\frac{cq_i^n cp_m}{Cc_m}\right)^{1/2}}{T_i}$$

It means that grid application wants to pay $Pc_i^{j^*}$ to computing power j for the needed resource to execute grid jobs under the completion time constraint.

$$\frac{\partial L_1(Pe_i^l, Pc_i^j, Pn_i^k)}{\partial Pn_i^k} = -1 + \frac{bq_i^n np_k}{Cn_k(Pn_i^k)^2} + er_i^n \frac{bq_i^n np_k}{Cn_k(Pn_i^k)^2} - \nu_i + \sigma_i \frac{bq_i^n np_k}{Cn_k(Pn_i^k)^2} + \varepsilon_i \frac{bq_i^n np_k}{Cn_k(Pn_i^k)^2}$$

Let $\partial L_1(Pe_i^l, Pc_i^j, Pn_i^k)/\partial Pn_i^k = 0$ to obtain

$$Pn_i^k = \left(\frac{(1 + er + \sigma_i + er.\varepsilon_i)bq_i^n np_k}{(1 + \nu_i)Cn_k}\right)^{1/2}$$

Using this result in the constraint equation, we can determine $\tau = (1 + er + \sigma_i + er.\varepsilon_i)/1 + \nu_i$ as

$$(\tau)^{-1/2} = \frac{T_i}{\sum_{m=1}^{N} \left(\frac{np_m bq_i^n}{Cn_m}\right)^{1/2}}$$

We obtain $Pn_i^{k^*}$

$$Pn_i^{k^*} = \left(\frac{bq_i^n np_k}{Cn_k}\right)^{1/2} \frac{\sum_{m=1}^{N} \left(\frac{bq_i^n np_m}{Cn_m}\right)^{1/2}}{T_i}$$

It means that grid application wants to pay $Pn_i^{k^*}$ to network resource k for the needed resource to execute grid jobs under the completion time constraint.

In problem (3.10), different resource providers compute the optimal resource allocation for maximizing the revenue of their own under constraints of resource capacity Ce_l, Cc_j, Cn_k. The objective of resource providers is to maximize $Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k$ under the constraints of their resource capacity.

We take the derivative and second derivative with respect to x_i:

$$U'_{\text{resource}}(e_i^l) = \frac{Pe_i^l}{e_i^j}, \qquad U''_{\text{resource}}(e_i^l) = -\frac{Pe_i^l}{e_i^{l2}}$$

$U''_{\text{resource}}(e_i^l) < 0$ is negative due to $0 < e_i^l$. The extreme point is the unique value maximizing the revenue of energy provider. The Lagrangian for (3.10) is $L_2(e_i^l, x_i^j, y_i^k)$.

$$\begin{aligned} L_2(e_i^l, x_i^j, y_i^k) &= \sum \left(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k\right) \\ &\quad + \lambda_i \left(Ce_l - \sum_i e_i^l\right) + \beta_i \left(Cc_j - \sum_i x_i^j\right) + \varphi_i \left(Cn_k - \sum_i y_i^k\right) \\ &= \sum \left(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k - \lambda_i e_i^l - \beta_i x_i^j - \varphi_i y_i^k\right) \\ &\quad + \lambda_i Ce_l + \beta_i Cc_j + \varphi_i Cn_k \end{aligned}$$

where λ_i, β_i and φ_i, is the Lagrangian constant. From Karush–Kuhn–Tucker theorem, we know that the optimal solution is given $\partial L_2(e_i^l, x_i^j, y_i^k)/\partial e_i^l = 0$ for $\lambda_i, \beta_i, \varphi_i > 0$.

Let $\partial L_2(e_i^l, x_i^j, y_i^j)/\partial e_i^l = 0$ to obtain

$$e_i^l = \frac{Pe_i^l}{\lambda_i}$$

Using this result in the constraint equation $Ce_l \geq \sum e_i^l$, we can determine λ_i as

$$\lambda_i = \frac{\sum_{d=1}^{n} Pe_i^d}{Ce_l}$$

We substitute λ into e_i^l to obtain

$$e_i^{l*} = \frac{Pe_i^l Ce_l}{\sum_{d=1}^n Pe_i^d}$$

e_i^{l*} is the unique energy allocation for maximizing the revenue of energy provider l.

Using the similar method, we can solve computing power allocation optimization problem:

$$U'_{\text{resource}}\left(x_i^j\right) = \frac{Pc_i^j}{x_i^j}, \qquad U''_{\text{resource}}\left(x_i^j\right) = -\frac{Pc_i^j}{x_i^{j2}}$$

$U''_{\text{resource}}(x_i^j) < 0$ is negative due to $0 < x_i^j$. The extreme point is the unique value maximizing the revenue of computing power provider.

Let $\partial L_2(e_i^l, x_i^j, y_i^k)/\partial x_i^j = 0$ to obtain

$$x_i^j = \frac{Pc_i^j}{\beta_i}$$

Using this result in the constraint equation $Cc_j \geq \sum x_i^j$, we can determine β_i as

$$\beta_i = \frac{\sum_{d=1}^n Pc_i^d}{Cc_j}$$

We substitute β into x_i^j to obtain

$$x_i^{j*} = \frac{Pc_i^j Cc_j}{\sum_{d=1}^n Pc_i^d}$$

x_i^{j*} is the unique optimal computing power allocation for maximizing the revenue of computing power provider j.

Using the similar method, we can solve network resource allocation optimization problem.

Let $\partial L_2(e_i^l, x_i^j, y_i^k)/\partial y_i^k = 0$ to obtain

$$y_i^k = \frac{Pn_i^k}{\varphi_i}$$

Using this result in the constraint equation $Cn_k \geq \sum y_i^k$, we can determine φ_i as

$$\varphi_i = \frac{\sum_{d=1}^n Pn_i^d}{Cn_k}$$

We substitute φ into y_i^k to obtain

$$y_i^{k*} = \frac{Pn_i^k Cn_k}{\sum_{d=1}^n Pn_i^d}$$

y_i^{k*} is the unique optimal network resource allocation for maximizing the revenue of network resource provider k.

4 Algorithm description

4.1 Mobile grid energy–QoS tradeoff algorithm (EQTA)

The objective of energy–QoS tradeoff scheduling is to maximize the utility of the grid system without exceeding the resource capacity, the energy budget, expense budget, and the deadline. The proposed algorithm decomposes energy–QoS tradeoff optimization problem into a sequence of two subproblems via an iterative algorithm. In each iteration, in subproblem 1, the grid application computes the unique optimal payment to resource provider under the energy budget, expense budget, and the deadline constraint to maximize the grid application's satisfaction. The grid application individually solves its fees to pay for energy resources, computing power, and network resource to complete its all jobs, adjusts its grid resource demand and notifies the grid resource provider about this change. In subproblem 2, different resource providers compute optimal resource allocation for maximizing the revenue of their own. Grid resource provider updates its price according to optimal payments from grid application, and then sends the new prices to the grid applications and allocates the resource for grid application, and the cycle repeats. The iterative algorithm that achieves mobile grid energy–QoS tradeoff optimization is described as follows.

Algorithm 1 Mobile Grid Energy–QoS Tradeoff Algorithm (EQTA)

```
Grid application $i$ behavior
Receives the new price $ep_l$ from the energy provider $l$;
    $Pe_i^{l*} = \mathrm{Max}\{U_{\mathrm{app}}(Pe_i^l, Pc_i^j, Pn_i^k)\}$;
    // Calculates $Pe_i^{l*}$ to maximize $U_{\mathrm{app}}(Pe_i^l, Pc_i^j, Pn_i^k)$
If $B_i \ge \sum_j Pc_i^j + \sum_k Pn_i^k + \sum_l Pe_i^l$
    Then Return $Pe_i^{l*}$ to energy resource $l$;
Else Return Null;
Receives the new price $cp_j$ from the computing power $j$;
    $Pc_i^{j*} = \mathrm{Max}\{U_{\mathrm{app}}(Pe_i^l, Pc_i^j, Pn_i^k)\}$;
    // calculates $Pc_i^{j*}$ to maximize $U_{\mathrm{app}}(Pe_i^l, Pc_i^j, Pn_i^k)$
If $B_i \ge \sum_j Pc_i^j + \sum_k Pn_i^k + \sum_l Pe_i^l$
    Then Return $Pc_i^{j*}$ to computing power $j$;
Else Return Null;
Receives the new price $np_k$ from the network resource provider $k$;
    $Pn_i^{k*} = \mathrm{Max}\{U_{\mathrm{app}}(Pe_i^l, Pc_i^j, Pn_i^k)\}$;
    // Calculates $Pn_i^{k*}$ to maximize $U_{\mathrm{app}}(Pe_i^l, Pc_i^j, Pn_i^k)$
If $B_i \ge \sum_j Pc_i^j + \sum_k Pn_i^k + \sum_l Pe_i^l$
    Then Return$Pn_i^{k*}$ to network resource $k$;
Else Return Null.
```

```
Computing power $j$, network resource $k$ and energy resource $l$
Receives optimal payments $Pe_i^{l^*}$, $Pc_i^{j^*}$, $Pn_i^{k^*}$ from grid application $i$;
If $Ce_l \geq \sum_{i=1}^{I} e_i^l$
Then
    $e_i^{l(n+1)*} = \mathrm{Max}\{U_{\mathrm{resource}}(e_i^l, x_i^j, y_i^k) = \sum_{i=1}^{I}(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k)\}$;
    // Calculates its optimal energy resource $e_i^{l(n+1)*}$
    $ep_l^{(n+1)} = \max\{\varepsilon, ep_l^{(n)} + \eta(e^l ep_l^{(n)} - Ce_l)\}$;    // Computes a new price
    // $e^l = \sum_{i=1}^{I} e_i^l$, $\eta > 0$ is a small step size parameter, $n$ is iteration number.
    Return energy resource price $ep_l^{(n+1)}$ to all grid applications;
Else Return Null;
If $Cc_i \geq \sum_{i=1}^{I} x_i^j$
Then
$x_i^{j(n+1)*} = \mathrm{Max}\{U_{\mathrm{resource}}(e_i^l, x_i^j, y_i^k) = \sum_{i=1}^{I}(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k)\}$;
    // Calculates its optimal computing power $x_i^{j(n+1)*}$
    $cp_j^{(n+1)} = \max\{\varepsilon, cp_j^{(n)} + \eta(x^j cp_j^{(n)} - Cc_j)\}$;    // Computes a new price
    // $x^j = \sum_i x_i^j$, $\eta > 0$ is a small step size parameter, $n$ is iteration number
    Return computing power price $cp_j^{(n+1)}$ to all grid applications;
Else Return Null;
If $Cn_k \geq \sum_{i=1}^{I} y_i^k$
Then
$y_i^{k(n+1)*} = \mathrm{Max}\{U_{\mathrm{resource}}(e_i^l, x_i^j, y_i^k) = \sum_{i=1}^{I}(Pe_i^l \log e_i^l + Pc_i^j \log x_i^j + Pn_i^k \log y_i^k)\}$;
    // Calculates its optimal network resource demand $y_i^{k(n+1)*}$
    $np_k^{(n+1)} = \max\{\varepsilon, np_k^{(n)} + \eta(y^k np_k^{(n)} - Cn_k)\}$;    // Computes a new price
    // $y^k = \sum_i y_i^k$, $\eta > 0$ is a small step size parameter, $n$ is iteration number
    Return network resource price $np_k^{(n+1)}$ to all grid applications;
Else Return Null.
```

4.2 Proof of algorithm convergence

The convergence of our mobile grid energy–QoS tradeoff algorithm (EQTA) can be proved through the Theorem 1. The following Theorem 1 shows that our algorithm converges to the Nash equilibrium solution [18]. Let V be a real valued function defined on R^J as follows:

$$V(cp) = \frac{\sum_j (cp_j - \overline{cp_j})^2}{2\eta}, \qquad V(ep) = \frac{\sum_l (ep_l - \overline{ep_l})^2}{2\eta},$$

$$V(np) = \frac{\sum_k (np_k - \overline{np_k})^2}{2\eta}$$

where $\overline{cp_j}$, $\overline{ep_l}$, $\overline{np_k}$ are the set of equilibrium prices of computing power j, energy resource l, network resource k. We consider V as a candidate Lyapunov function associated with the subsequence $\{cp_j^{(n)}, ep_l^{(n)}, np_k^{(n)}\}$. Notice that V is convex, bounded, and differentiable. The algorithm is defined by a positive constant step size.

Theorem 1 *Let* $\{cp_j^{(n)}, ep_l^{(n)}, np_k^{(n)}\}$ *be a sequence generated by the above algorithm for an arbitrary initial value* $cp_j^{(0)}, ep_l^{(0)}, np_k^{(0)} \in R^J$ *and* $0 < \eta < 1$, *then* $\{cp_j^{(n)}, ep_l^{(n)}, np_k^{(n)}\}$ *converges to* $\overline{cp_j}, \overline{ep_l}, \overline{np_k}$.

Proof The first partial derivatives of $V(cp)$can be easily obtained.

$$\frac{\partial V(cp_j)}{\partial cp_j} = \frac{(cp_j - \overline{cp_j})}{\eta} \tag{4.1}$$

∇V is Lipschitz continuous, that is,

$$\left\| \nabla V(cp_j) - \nabla V(cp_{j+1}) \right\| \le L \| cp_j - cp_{j+1} \| \tag{4.2}$$

By the definition of the algorithm, (4.1)

$$\nabla cp_j = cp_j^{(n+1)} - cp_j^{(n)} = \eta \left(\sum_i x_i^j - Cc_j \right) \tag{4.3}$$

let $L = 1$. From (4.2) by the descent lemma [18, Appendix A.24], the following holds:

$$\begin{aligned} V\left(cp_j^{(n+1)}\right) &\le V\left(cp_j^{(n)}\right) + \nabla cp_j \nabla V\left(cp_j^{(n)}\right) + \|\nabla cp_j\|^2 \\ &= V\left(cp_j^{(n)}\right) + \left(\frac{(cp_j^{(n)} - \overline{cp_j})}{\eta} + \nabla cp_j \right) \nabla cp_j \end{aligned} \tag{4.4}$$

Consider the following two cases:

(1) $\overline{cp_j} > cp_j^{(n+1)} > cp_j^{(n)} \cdot \nabla cp_j = cp_j^{(n+1)} - cp_j^{(n)} > 0$, $cp_j^{(n)} - \overline{cp_j} < 0$. Then from (4.4)

$$\begin{aligned} V\left(cp_j^{(n+1)}\right) &\le V\left(cp_j^{(n)}\right) + \left(\frac{(cp^{(n)} - \overline{cp})}{\eta} + \nabla cp_j \right) \nabla cp_j \\ &= V\left(cp_j^{(n)}\right) + \frac{1}{\eta} \left((1-\eta) cp_j^{(n)} - \overline{cp} + \eta cp_j^{(n+1)} \right) \nabla cp_j \\ &\le V\left(cp_j^{(n)}\right) + \frac{1}{\eta} \left(cp_j^{(n)} - \overline{cp} \right) \nabla cp_j \end{aligned}$$

From the assumption $0 < \eta < 1$, the second term on the right-hand side of (4.4) is negative,

$$V\left(cp_j^{(n+1)}\right) \le V\left(cp_j^{(n)}\right)$$

(2) $cp_j^{(n)} > cp_j^{(n+1)} > \overline{cp_j} \nabla cp_j = cp_j^{(n+1)} - cp_j^{(n)} < 0,\ cp_j^{(n)} - \overline{cp_j} > 0$. Then from (4.4)

$$\begin{aligned} V\big(cp_j^{(n+1)}\big) &\le V\big(cp_j^{(n)}\big) + \left(\frac{(cp^{(n)} - \overline{cp_j})}{\eta} + \nabla cp_j\right)\nabla cp_j \\ &\le V\big(cp_j^{(n)}\big) + \big(cp_j^{(n)} - \overline{cp_j} + cp_j^{(n+1)} - cp_j^{(n)}\big)\nabla cp_j \end{aligned} \tag{4.5}$$

From the assumption $0 < \eta < 1$, the second term on the right-hand side of (4.5) is negative,

$$V\big(cp_j^{(n+1)}\big) \le V\big(cp_j^{(n)}\big)$$

Given $V(cp_j^{(n+1)}) \le V(cp_j^{(n)})$ for all n, $\|\nabla V(cp_j^{(n)})\| \to 0$ as n goes to ∞. Since $V(cp)$ is convex on R^J and has a unique global minimum [18], then $\{cp \in R^J | V(cp) < V(cp^{(0)})\}$ is compact. Therefore, $\{cp_j^{(n)}\}$ is bounded and it has at least one limit point. Let $\overline{cp_j}$ be one limit point. $\overline{cp_j}$ is then a stationary point of $V(cp)$ since $\nabla V(cp)$ is continuous and $\lim_{n\to\infty} \nabla V(cp^{(n)}) = 0$.

Using the similar method, we can prove $\{ep_l^{(n)}, np_k^{(n)}\}$ converge to $\overline{ep_l}$, $\overline{np_k}$. □

5 Simulations and analysis

5.1 Simulation environments

In this section, we present the performance evaluation of our energy consumption and QoS tradeoff optimization algorithm (EQTA). Our simulator supports a topology of multiple LANs connected through wired nodes and wireless LANs and bandwidth monitoring. The proposed simulator considers mobile grid environment parameters such as the battery (power) state, the network state (latency and bandwidth), and the system loading state (CPU and memory). The grid simulator is implemented on top of the JAVASIM network simulator [19]. In order to simulate the dynamics and heterogeneity of the grid, all values of networks can be changed after topology generation. Network generator BRITE [20] generates the computer network topology. BRITE is a random network topology generator used to generate the simulation test bed. In the simulator, different agents are used, namely resource provider agents, user agents, and grid scheduler agents which implements the EQTA algorithm. The grid scheduler receives the task request, schedules the tasks to the host nodes, and then writes the scheduling records to the files for statistical analysis. The grid scheduler starts a listening thread that listens to the task requests. It receives the task requirements and puts them into the task queue. While the task queue is not empty, the grid scheduler starts the scheduling algorithm to find the right match. When resource agent updates its price, the resource agent forwards the price to user agents; the resource price is put in a packet. Whenever the new price packet passes to the user agent, the user agent calculates the utility. According to the algorithm, if the price becomes higher than its maximum willingness to pay, the user agent does not buy grid resources. The user agent can be informed of the price for the next iteration by the next price packets.

We simulate a grid environment containing 10 grid domains. To simulate grid domain, we profile each node in a domain with a group of parameters to represent arrival rate, machine computing power, energy state, and communication bandwidth, respectively. We assume that each grid application can use any of the grid resources including computation, communication, and energy resources. Processor capacity varies from 220 to 580 MIPS. The wireless network bandwidth is from 10 Kbps to 1 Mbps. The main memory is set by 128 M, 256 M, 512 M, and 2 G. The disk capacity is set by 80 G, 30 G, and 20 G. The selective grid applications for simulation are computation-intensive applications such as image processing applications and mpeg players. The simulator leaves each application on the mobile device or delegates it to a grid node. There are a total of 150 resources and 600 applications are taken for experimental evaluation of the system. Energy consumption is represented as a percentage of the total energy required to meet all job deadlines. Assume that the maximum power, $P_{\max}$, corresponds to running all jobs with the maximum processing frequency. The maximum frequency is assumed to be $f_{\max} = 1$ and the maximum frequency-dependent power is $P_{\max} = 1$. When the energy budget for each interval is limited, we can only consume a fraction of $P_{\max}$ when processing requests during a given interval. Jobs arrive at each site s_i, $i = 1, 2, \ldots, n$, according to a Poisson process with rate α. To take into account the wide dispersion in the job sizes in real grid applications, the sizes of the jobs are taken randomly from the uniform distribution in the interval [1, 100]. The capacities of the resources were also chosen uniformly in the interval [50, 500]. The resource cost can be expressed in grid dollar that can be defined as unit processing cost. The initial price of resource is set from 10 to 500 grid dollars. Users submit their jobs with varying deadlines. The deadlines

Table 2 Simulation parameters

Simulation parameter	Value
Number of domains	10
Number of nodes in each domains	50
Bandwidth	[10 Kbps, 1 Mbps]
Network transferring time	200 ms
Processor capability (MIPS)	[220, 580]
RAM (B)	[128 M, 256 M, 512 M, 2 G]
Hard disk (B)	[80 G, 30 G, 20 G]
Arrival time	[100 ms, 400 ms]
Total number of applications	600
Total number of resource providers	150
Reschedule interval	600 ms
Initial price of computing power (grid dollar)	[10, 500]
Deadline	[100, 400]
Expense budget	[100,1500]
Energy budget	[0.1, 1.0]
Resource capacities	[50, 500]
Job arrival rate	[0.1, 0.6]

Table 3 Description of the grid users

Grid users	Deadline	Budget	Bandwidth (Kbps)		Processor capacity (MIPS)		Energy consumption	
			Min	Max	Min	Max	Min	Max
1	100	1500	100	500	220	410	0.3	0.6
2	200	500	10	500	220	370	0.3	0.6
3	300	100	10	100	220	270	0.2	0.5
4	400	500	10	800	340	390	0.2	0.4
5	100	1000	100	1000	340	510	0.2	0.5
6	400	500	10	500	220	270	0.1	0.3
7	200	1500	500	1000	340	390	0.2	0.4
8	100	1000	100	1000	220	370	0.1	0.3
9	300	1500	100	800	340	510	0.3	0.6
10	200	100	200	500	220	410	0.2	0.4

of users are chosen from 100 ms to 400 ms. The budgets of users are set from 100 to 1500 grid dollars. Each measurement is run 6 times with different seeds. Simulation parameters are listed in Table 2. The descriptions of grid users are listed in Table 3.

5.2 Performance study of EQTA algorithm

The first series of experiments are conducted to evaluate our mobile grid energy–QoS tradeoff algorithm (EQTA) in terms of the utility, resource provider's revenue, energy consumption ratio, execution success ratio, and resource utilization under different system load. The revenue for providers (*RP*) is the sum of revenues for all of its resource types. Energy consumption ratio (*EC*) is defined as the percentage of consumed energy among total available energy resources. Execution success ratio (*ES*) is the percentage of tasks executed successfully before their deadline. Resource utilization (*RU*) is the ratio of the consumed resources to the total resources available as a percentage and commonly refers to the percent of time a resource is busy. System load is defined as the ratio of the total number of requests arrived in one interval over the number of requests that can be handled by the system within one interval. The value of system load expresses the extent to which the whole system is busy. If in a certain period of time the number of jobs submitted to the system is small, the system load is light; otherwise, the system load is heavy. System load influences the performance of scheduling inherently. Load factor (*LF*) varies from 0.1 to 0.9.

The first experiment is to test convergence of our mobile grid energy–QoS tradeoff algorithm (EQTA). For an iterative algorithm, the average number of convergence iterations is a very important factor, since an algorithm with smaller convergence iterations needs shorter scheduling time, and can achieve higher speed scheduling. Convergence iterations are tested under varying system load (*LF*). In Fig. 1, when the system load is low ($LF = 0.1$), the convergence iterations of EQTA decrease to one. When the system load increases to 0.6 ($LF = 0.6$), the convergence iterations of EQTA increase to two. When the system load increases to 0.9 ($LF = 0.9$), the convergence iterations of EQTA increase to six. In task intensive case ($LF = 0.9$), there

Fig. 1 Convergence iterations under various load factor

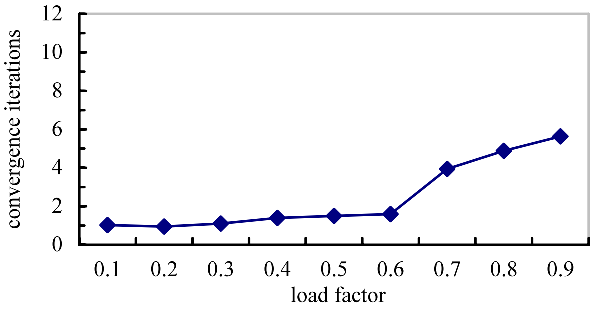
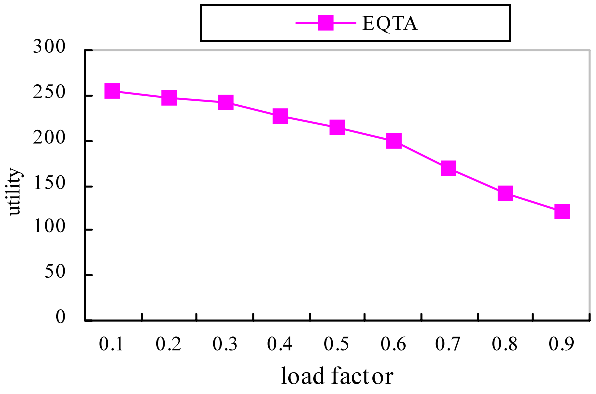


Fig. 2 Utility under various load factor



are more tasks waiting for a match than the numbers of resources. As the demand is more than the supply, the price for the resources increases. When load factor increases quickly, the grid resource supply is not enough to be allocated to user, the unit price of the resource will increase, and some users will not afford the grid resources.

The following five figures (Figs. 2–6) are to study the utility, resource provider's revenue, execution success ratio, energy consumption ratio and resource utilization of EQTA scheduling algorithm under different load factor respectively. Load factor varies from 0.1 to 0.9. Figure 2 shows the effect of load factor on the utility. The smaller is LF, the higher is the user utility. When $LF = 0.9$, the user utility is as much as 47% less than that by $LF = 0.10$. The value of LF is low, the system is lightly loaded, and the unit price of the resource is cheap; user application can choose cheap resources to complete tasks under the deadline, so the benefit of the user application (the utility) is high. When the system is heavily loaded, the unit price of the resource is expensive; the benefit of the user application (the utility) is less. Figure 3 shows the effect of load factor on the revenue of grid resource provider. Before the load factor reaches 0.7 ($LF = 0.7$), the revenue increases as the load factor increases. After the load factor exceeds 0.7, the revenue decreases when the load factor increases. This is easily understood. When $LF < 0.7$, the grid resource supply is enough to be allocated to the user, so the unit price of the resource is cheap; more users can choose resources to complete tasks, so the resource providers will get more revenue from grid user applications. When $LF > 0.7$, the unit price of the resource is expensive; some applications with low expense budgets cannot afford to choose proper resources to complete

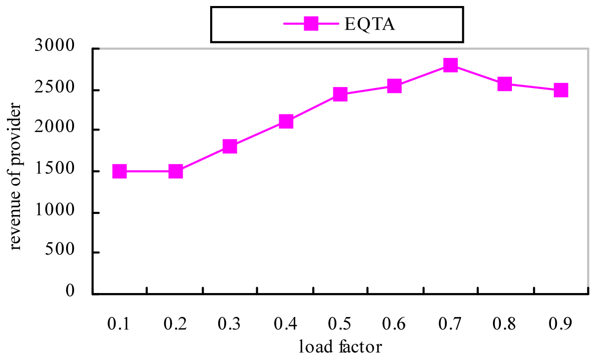


Fig. 3 Revenue of provider under various load factor

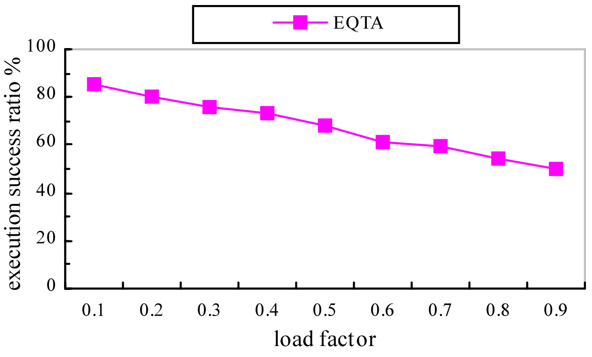


Fig. 4 Execution success ratio under various load factor

their tasks, so the resource providers will get less revenue from grid user applications. Considering the execution success ratio, from the results in Fig. 4, when the load factor is 0.5 ($LF = 0.5$), the execution success ratio is 22% more than $LF = 0.9$. When load factor increases, execution success ratio deteriorates quickly. When $LF = 0.7$, the execution success ratio is as much as 31% less than that with $LF = 0.1$. When the load factor increases, fewer user applications can be admitted into the system due to the increase of system burden, so, fewer applications can be executed successfully before their deadline. Figure 5 shows the energy consumption ratio under different system loads. When system load increases, more requests need to be processed within one interval and the energy consumption ratio increases. When increasing the load factor by $LF = 0.7$, the energy consumption ratio of EQTA is as much as 17% more than $LF = 0.4$. Figure 6 shows as load factor increases, the resource utilization ratio increases. When $LF = 0.8$, the resource utilization of EQTA is as much as 19% more than utilization by $LF = 0.4$. When the load factor increases, many tasks are sent to the system, grid resources are busier, and the resource utilization is high.

5.3 Comparison experiments

The experiments are conducted to compare our mobile grid energy–QoS tradeoff algorithm (EQTA) under the constraint of the resource capacity, the energy budget, expense budget, and the deadline with low-energy earliest deadline-first (LEDF)

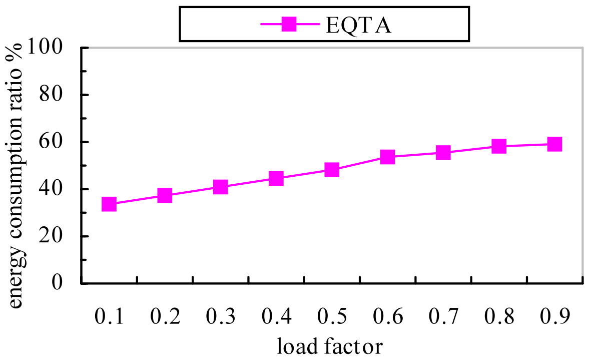


Fig. 5 Energy consumption ratio under various load factor

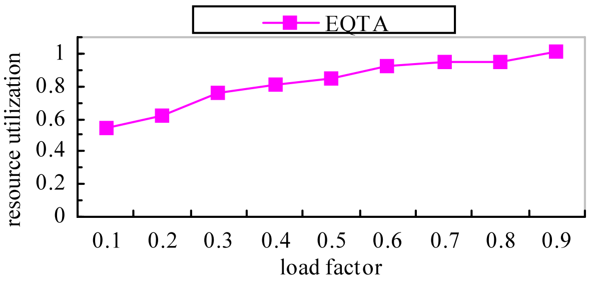


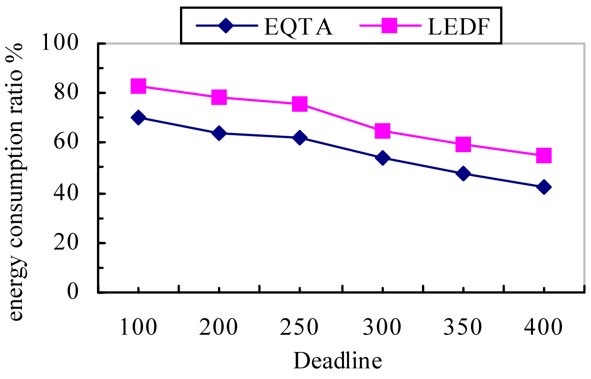
Fig. 6 Resource utilization under various load factor

scheduling algorithm [21] proposed by Swaminathan and Chakrabarty. The reason for choosing LEDF as the comparison is that both our work and LEDF deal with energy and QoS constrained scheduling. Swaminathan and Chakrabarty [21] studied scheduling workloads containing periodic tasks in real-time systems. The proposed approach minimizes the total energy consumed by the task set and guarantees that the deadline for every periodic task is met. They present a mixed-integer linear programming model for the NP-complete scheduling problem. They proposed a low-energy earliest deadline-first (LEDF) scheduling algorithm. The operation of the low-energy earliest deadline first (LEDF) is as follows:

1. LEDF maintains a list of all released tasks, called the *ready list*.
2. When tasks are released, the task with the nearest deadline is chosen to be executed.
3. A check is performed to see if the task deadline can be met by executing it at the lower voltage (speed).
4. If the deadline can be met, LEDF assigns the lower voltage to the task and the task begins execution.
5. During the task's execution, other tasks may enter the system. These tasks are placed automatically on the *ready list*.
6. LEDF again selects the task with the nearest deadline to be executed. As long as there are tasks waiting to be executed, LEDF does not keep the processor idle.
7. This process is repeated until all the tasks have been scheduled.

In the simulation, we compare mobile grid energy–QoS tradeoff algorithm (EQTA) with low-energy earliest deadline-first (LEDF) scheduling algorithm by bud-

Fig. 7 Energy consumption ratio under various deadline



get and job deadline to study how they affect the performance of two algorithms. The performance metrics include energy consumption ratio, resource utilization, deadline miss ratio, and allocation efficiency. The job expense budget (B) is set from 100 to 1500. The job deadline (T) is set from 100 to 400. Deadline Miss Ratio (*DM*) is defined as the ratio of the number of jobs whose deadline constraints are not met over the total number of jobs. Energy consumption ratio (*EC*) is defined as the percentage of consumed energy among total available energy resources. Resource utilization ratio (*RU*) is defined as the percentage of allocated resources among all available resources. Allocation Efficiency (*AE*) is a measure of the efficiency of the allocation process, which is computed using the number of all requests and number of accepted requests.

How the deadline affects energy consumption ratio, resource utilization, deadline miss ratio and allocation efficiency were illustrated in Figs. 7–10, respectively. Figure 7 shows the energy consumption ratio varies under different deadlines. From the results in Fig. 7 when the deadline is low, there is intensive demand for the resources in short time, so the user application should choose more energy-consuming resources to process the jobs, and the energy consumption ratio is high. However, when the deadline changes to higher, it is likely that jobs can be completed before the deadline, so grid job considers using the energy saving resources to complete tasks to maximize the utility, then the energy consumption ratio is low. When the deadline is 100 ($T = 100$), energy consumption ratio of LEDF is 82%; energy consumption ratio of EQTA is 70%. Compared with LEDF, EQTA consumes less energy especially when the deadline is low. For the resource utilization under different deadline constraints, from the results in Fig. 8, when increasing the deadlines, the impact on the resource utilization is obvious. A larger deadline values brings out higher resource utilization. When the deadline is 400 ($T = 400$), the resource utilization of EQTA is 19% higher than LEDF. Under the same deadline, EQTA has higher resource utilization than LEDF. Because within certain deadline, grid jobs in EQTA can choose resources of different prices to maximize the grid application utility; so can achieve good resource utilization. Figure 9 is to show the effect of the deadline on deadline miss ratio. When the deadline is low, deadline miss ratios of LEDF and EQTA are high. When increasing deadline, deadline miss ratios of two schemes become lower. Because of low deadlines, more jobs cannot be completed on time. When the dead-

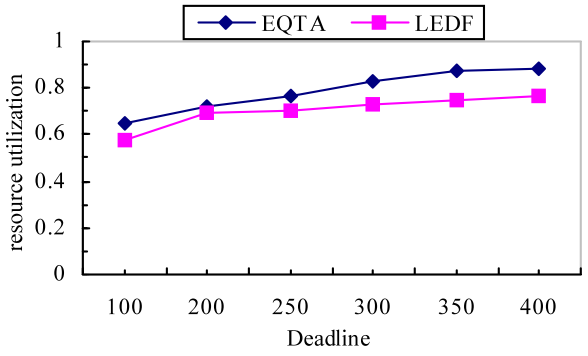


Fig. 8 Resource utilization under various deadline

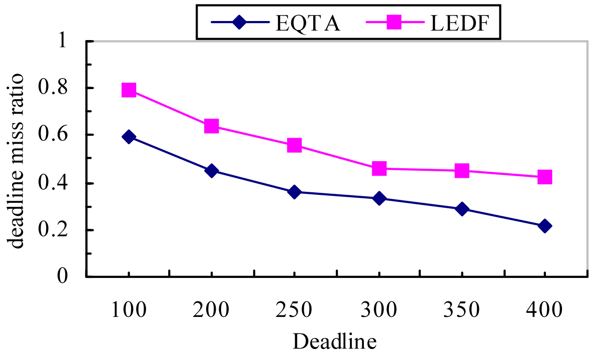


Fig. 9 Deadline miss ratio under various deadline

line is 100 ($T = 100$), the deadline miss ratio of LEDF increases to 80%; deadline miss ratio of EQTA increases to 59%. From the results in Fig. 10, the allocation efficiency increases when the deadline increases. When the deadline is low, there is intensive demand for the resources in short time; some user's requirements cannot be processed on time. The job with a low budget cannot be completed before deadline; this leads to low allocation efficiency. When the deadline is 400 ($T = 400$), the allocation efficiency of EQTA is 50% higher than $T = 100$. Compared with LEDF, the allocation efficiency of EQTA decreases slowly than LEDF when the deadline decreases. When deadline is 100 ($T = 100$), allocation efficiency of LEDF decreases to 37%, allocation efficiency of EQTA decreases to 54%.

Figures 11–14 illustrated the effects of the user expense budget on the energy consumption ratio, resource utilization, deadline miss ratio, and allocation efficiency. The grid application budget (B) is set from 100 to 1500. Figure 11 shows the energy consumption ratio under different expense budgets. A larger budget enables grid user application to use more energy resources with high price to complete the task before its deadline. When expense budgets are high, energy consumption ratio is high. When $B = 1000$, the energy consumption ratio of EQTA is 40% more than energy consumption ratio by $B = 100$. Compared with LEDF, the energy consumption ratio of EQTA decreases slowly than LEDF when the expense budget decreases. When expense budget is 800 ($B = 800$), energy consumption ratio of LEDF increases to 79%, energy consumption ratio of EQTA is up to 67%. Considering the resource uti-

Fig. 10 Allocation efficiency under various deadline

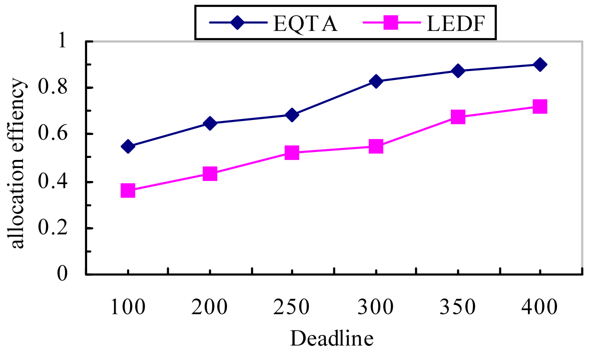


Fig. 11 Energy consumption ratio under various budget

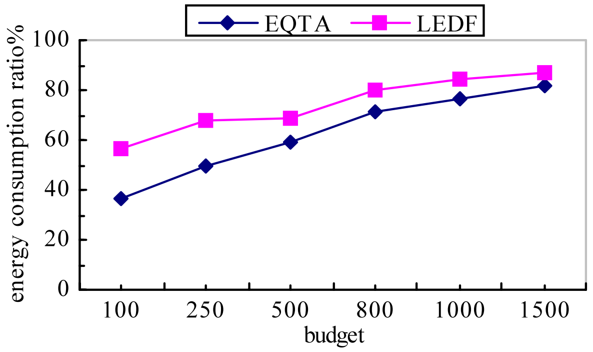
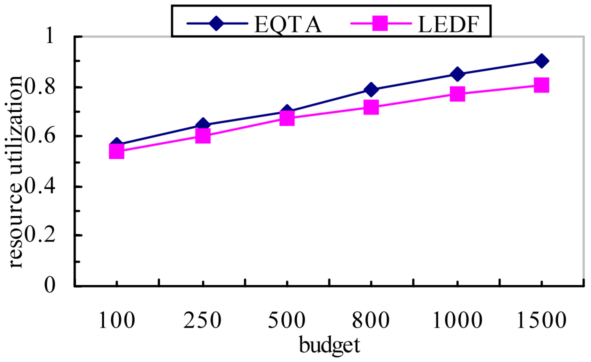


Fig. 12 Resource utilization under various budget



lization, from the results in Fig. 12, as the expense budget is higher, the resource utilization becomes higher. When $B = 1000$, the resource utilization of EQTA is as much as 38% more than resource utilization by $B = 100$. Because when the budget decreases quickly, the users will be prevented from obtaining expensive resources. When expense budget is 250 ($B = 250$), the resource utilization of LEDF decreases to 57%, the resource utilization of EQTA decreases to 64%. Figure 13 is to show the effect of an expense budget on the deadline miss ratio. When increasing budget values, the deadline miss ratio becomes lower. A larger expense budget enables grid user

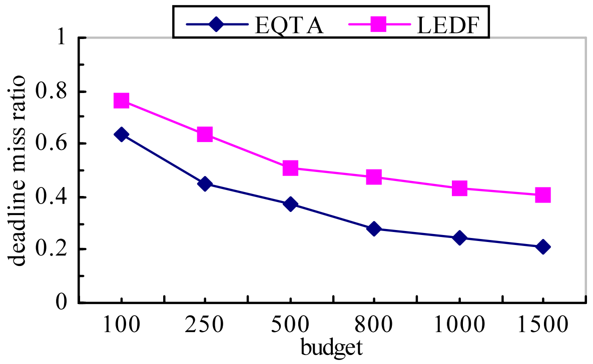


Fig. 13 Deadline miss ratio under various budget

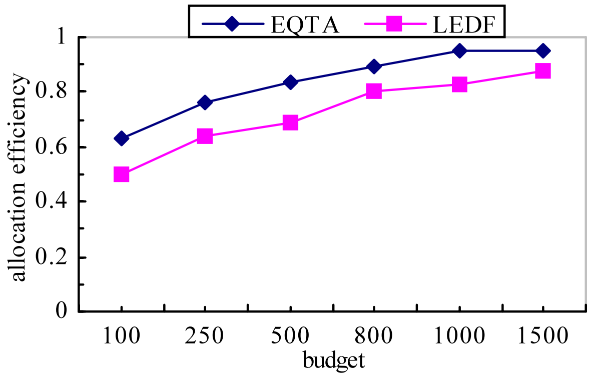


Fig. 14 Allocation efficiency under various budget

to afford more expensive resources to complete the task before its deadline. When the budget increases ($B = 1500$), the deadline miss ratio of EQTA is as much as 48% less than that with $B = 100$. Under the same expense budget ($B = 800$), EQTA has 29% lower deadline miss ratio than LEDF. Considering the allocation efficiency, from the results in Fig. 14, when increasing budget values by $B = 1000$, the allocation efficiency of EQTA is 42% more than that by $B = 100$. Under the same budget value ($B = 1000$), the allocation efficiency of EQTA becomes 32% higher than LEDF. A larger budget brings out higher allocation efficiency. Because grid applications can use expensive resources to complete the tasks within the deadline, they can maximize their allocation efficiency.

Our energy and QoS tradeoff algorithm jointly considers both the benefits of the grid resource and grid application. In the EQTA algorithm, we consider two aspects of the utility function: one is the utility function for maximizing benefits of grid applications and the other is for optimizing revenue of resources. LEDF is an application-centric scheduling, which mainly considers user benefit. The objective of LEDF is to minimize the total battery energy used to successfully accomplish a task, as well as optimize time deadline. Resource utilization and allocation efficiency are performance metrics for grid resource provider, energy consumption ratio and deadline miss ratio are performance metrics for grid user. So from the above results, the resource utilization and allocation efficiency of EQTA are better than LEDF; the energy consumption ratio and deadline miss ratio of EQTA are close to LEDF.

6 Considerations for integration into EGEE

Most current real grid environments do not consider market-based schemes for integration of mobile devices into mobile grids. We will consider moving our method to grid infrastructures such as EGEE [22] to test its feasibility. But the real grid environment is more complex, and integration of our multiple granularity control algorithms into EGEE is not a simple task. EGEE grid is one of the largest multidisciplinary grid infrastructures in the world. The EGEE infrastructure is federating resources and making them easily accessible but does not own the resources itself. Instead, the resources belong to independent resource centers that procure their resources and allow access to them based on their particular policies. In order to implement our method into EGEE, we will integrate energy and the QoS balancing algorithm in mobile grid with gLite.

The gLite middleware deployed on the EGEE infrastructure integrates the sites' computing resources through the workload management system (WMS). The WMS is a set of middleware level services responsible for the distribution and management of jobs. The core of the WMS is the workload manager which accepts jobs from users and dispatches them to computational resources based on the users' requirements on one hand, and the characteristics (e.g., hardware, software, localization) and state of the resources on the other hand. The workload manager is implemented as a distributed set of resource brokers; the brokers get a consistent view of the resource availability through the grid information system. Each broker reaches a decision by a matchmaking process between submission requests and available resources. In order to launch and control gLite jobs, the grid user requires a remote command execution and file copying mechanism between workstation and grid. This can be done by installing gLite client tools, using https, or remote grid UI node. Job requirements are described via the Job Description Language (JDL). Submission commands are called to submit gLite jobs. When a gLite job finishes, its output data and files are copied from the storage element. The user then retrieves the output of the application and destroys the job.

7 Conclusions

Energy aware grid scheduling present two challenges: QoS provisioning and energy saving at the same time. The paper presents a tradeoff policy between energy consumption and QoS in mobile grid environment. Utility function is specified for each QoS dimension; we model tradeoff policy between energy consumption and QoS by utility optimization. The work is different from the classical energy aware scheduling, which usually takes the consumed energy as constraints. Our utility model regards consumed energy as one of the components of measure of the utility values, which indicates the tradeoff of application satisfaction and consumed energy. It is a more accurate utility model for abstracting the energy characteristics for mobile users and resources in mobile grid. The paper proposes a distributed energy–QoS tradeoff algorithm. The performance evaluation of our energy–QoS tradeoff algorithm is evaluated and compared with other energy and deadline constrained scheduling algorithms. Because most current mobile grid environments do not consider energy–QoS tradeoffs at the current stage, we focus on how to solve the problem and test our approach by simulation test; secondly, we will consider moving our method to grid infrastructures such as EGEE.

Acknowledgements The authors thank the editor and the anonymous reviewers for their helpful comments and suggestions. The work is supported by National Natural Science Foundation of China (NSF) under grants (No. 60773211, No. 60970064), Program for New Century Excellent Talents in University, China (NCET-08-0806), and the Natural Science Foundation of HuBei Province under Grant (No. 2008CDB335), Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

Appendix: Lagrangian relaxation

Lagrangian relaxation is a relaxation technique which works by moving hard constraints into the objective so as to exact a penalty on the objective if they are not satisfied.

Mathematical description

Given a linear programming (LP) problem $x \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m,n}$ of the following form:

$$\begin{aligned} \max \quad & C^{\mathrm{T}} X \\ \text{s.t.} \quad & Ax \le b \end{aligned}$$

If we split the constraints in A, such that $A_1 \in \mathbb{R}^{m_1 \cdot n}$, $A_2 \in \mathbb{R}^{m_2 \cdot n}$ and $m_1 + m_2 = m$, we may write the system:

$$\begin{aligned} \max \quad & C^{\mathrm{T}} X \\ \text{s.t.} \quad & A_1 x \le b_1 && \text{(A.1)} \\ & A_2 x \le b_2 && \text{(A.2)} \end{aligned}$$

We may introduce the constraint (2) into the objective:

$$\begin{aligned} \max \quad & c^{\mathrm{T}} x + \lambda^{\mathrm{T}}(b_2 - A_2 x) \\ \text{s.t.} \quad & A_1 x \le b_1 && \text{(A.1)} \end{aligned}$$

If we let $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_{m_2})$be nonnegative weights, we get penalized if we violate the constraint (2), and we are also rewarded if we satisfy the constraint strictly. The above system is called the Lagrangian relaxation of our original problem.

Of particular use is the property that for any fixed set of $\tilde{\lambda}$ values, the optimal result to the Lagrangian relaxation problem will be no smaller than the optimal result to the original problem. Let $\hat{x}$ be the optimal solution to the original problem, and let $\bar{x}$ be the optimal solution to the Lagrangian relaxation. We can then see that

$$C^{\mathrm{T}}\hat{x} \leq c^{\mathrm{T}}\hat{x} + \tilde{\lambda}^{\mathrm{T}}(b_2 - A_2\hat{x}) \leq c^{\mathrm{T}}\bar{x} + \tilde{x}^{\mathrm{T}}(b_2 - A_2\bar{x})$$

The first inequality is true because $\hat{x}$ is feasible in the original problem and the second inequality is true because $\overline{x}$ is the optimal solution to the Lagrangian relaxation. This in turn allows us to address the original problem by instead exploring the partially dualized problem

$$\min \quad P(\lambda) \quad \text{s.t.} \quad \lambda \geq 0$$

where we define $P(\lambda)$ as

$$\begin{aligned} \max \quad & c^{\mathrm{T}}x + \lambda^{\mathrm{T}}(b_2 - A_2x) \\ \text{s.t.} \quad & A_1x \leq b_1 \end{aligned} \tag{A.1}$$

A Lagrangian relaxation algorithm thus proceeds to explore the range of feasible λ values while seeking to minimize the result returned by the inner P problem. Each value returned by P is a candidate upper bound to the problem, the smallest of which is kept as the best upper bound. If we additionally employ a heuristic, probably seeded by the $\overline{x}$ values returned by P, to find feasible solutions to the original problem, then we can iterate until the best upper bound and the cost of the best feasible solution converge to a desired tolerance.

References

1. Borges VCM, Dias JS, Rossetto AGM, Dantas MAR (2007) SuMMIT an architecture for mobile devices to coordinate the execution of applications. In: WETICE 2007: 16th IEEE international workshops on grid environments enabling technologies: infrastructure for collaborative enterprises, 18–20 June 2007, pp 217–222
2. Zong Z, Qin X (2007) Energy-efficient scheduling for parallel applications running on heterogeneous clusters. In: International conference on parallel processing (ICPP 2007). IEEE, New York
3. Huang Y, Mohapatra S, Venkatasubramanian N (2005) An energy-efficient middleware for supporting multimedia services in mobile grid environments. In: IEEE international conference on information technology
4. Park E, Shin H, Kim SJ (2007) Selective grid access for energy-aware mobile computing. In: Indulska J et al (eds) Proceedings of UIC 2007. LNCS, vol 4611. Springer, Berlin, pp 798–807
5. Hummel KA, Jelleschitz G (2007) A robust decentralized job scheduling approach for mobile peers in ad-hoc grids. In: CCGRID 2007: Seventh IEEE international symposium on cluster computing and the grid. May 2007, pp 461–470
6. Wong S-W, Ng K-W (2008) Performance evaluation of mobile grid services. In: Nguyen NT et al (eds) KES-AMSTA 2008. LNAI, vol 4953. Springer, Berlin, pp 557–566
7. AlEnawy TA, Aydin H (2005) Energy-constrained scheduling for weakly-hard real-time systems. In: Proceedings of the 26th IEEE international real-time systems symposium (RTSS'05). IEEE, New York
8. Xie T, Qin X, Nijim M (2006) Solving energy-latency dilemma: task allocation for parallel applications in heterogeneous embedded systems. In: Proceedings of the 2006 international conference on parallel processing (ICPP'06). IEEE, New York
9. Kim KH, Buyya R, Kim J (2007) Power aware scheduling of bag-of-tasks applications with deadline constraints on DVS-enabled clusters. In: Proceedings of the seventh IEEE international symposium on cluster computing and the grid. IEEE Computer Society, Washington, pp 541–548

10. Katsaros K, Polyzos GC (2007) Optimizing operation of a hierarchical campus-wide mobile grid for intermittent wireless connectivity. In: IEEE LAN/MAN workshop (LANMAN 2007)
11. Li C, Li L (2007) Utility based QoS optimisation strategy for multi-criteria scheduling on the grid. J Parallel Distributed Comput 67(2):142–153
12. Li C, Li L (2007) Joint QoS optimization for layered computational grid. Inf Sci 177(15):3038–3059
13. Li L, Li L (2004) Agent framework to support computational grid. J Syst Softw 70(1–2):177–187
14. Li L, Li L (2006) Multi-economic agent interaction for optimizing the aggregate utility of grid users in computational grid. Appl Intell 25(2):147–158
15. Li C, Li L (2005) A distributed utility-based two level market solution for optimal resource scheduling in computational grid. Parallel Comput 31(3–4):332–351
16. Luh PB, Hoitomt DJ (1993) Scheduling of manufacturing systems using the Lagrangian relaxation technique. IEEE Trans Autom Control 38(7):1066–1079
17. Kelly F, Maulloo A, Tan D (1998) Rate control for communication networks: shadow prices, proportional fairness and stability. J Oper Res Soc 49(3):237–252
18. Bertsekas D (1999) Nonlinear Programming, 2nd edn. Athena Scientific
19. JAVASIM, http://javasim.ncl.ac.uk
20. BRITE, http://www.cs.bu.edu/brite
21. Swaminathan V, Chakrabarty K (2001) Real-time task scheduling for energy-aware embedded systems. J Franklin Inst 338:729–750
22. EGEE, http://www.eu-egee.org/