MÔ PHỎNG THUẬT TOÁN LẬP LỊCH TRONG TÍNH TOÁN LƯỚI

SIMULATION of SCHEDULING ALGORITHM IN GRID COMPUTING

Trần Hồ Thủy Tiên

Trường Đại học Bách Khoa, Đại học Đà Nẵng; thttien@dut.udn.vn

Tóm tắt - Các Cluster, Grid, và P2P nỗi bật lên như các mô hình phổ biến cho tính toán song song và phân tán. Chúng có khả năng kết hợp các tài nguyên phân tán để giải quyết các vấn đề trên diện rộng trong khoa học, kỹ thuật và tài chính. Trong các môi trường tính toán Grid và P2P, các tài nguyên thường phân tán về mặt địa lý trong các miền điều hành, quản trị và sở hữu của các tổ chức khác nhau với các chiến lược khác nhau được kết nối bằng WAN hay Internet. Quản lý các tài nguyên và lập lịch các ứng dụng trong các hệ phân tán trên diện rộng là một nhiệm vụ phức tạp. Cần phải cải tiến một cách hiệu quả các bộ môi giới tài nguyên và các thuật toán lập lịch. Bài báo giới thiệu thuật toán lập lịch mang tính kinh tế trong tính toán lưới sử dụng công cụ mô phỏng Grid dựa vào sự kiện rời rạc Java, gọi là GridSim. Một công cụ mô hình hoá và mô phỏng tài nguyên Grid, các user, các mô hình ứng dụng.

Từ khóa - Tính toán lưới; Mô phỏng lưới; Tài nguyên; Bộ môi giới tài nguyên; Các phần tử xử lý.

Abstract - Clusters, Grids, and peer-to-peer (P2P) networks have emerged as popular paradigms for next generation parallel and distributed computing. They can aggregate distributed resources to solve large-scale problems in science, engineering, and commerce. In Grid and P2P computing environments, the resources are usually geographically distributed in multiple administrative domains managed and owned by different organizations with different policies, and interconnected by wide-area networks or the Internet. The management of resources and scheduling of applications in such large-scale distributed systems is a complex undertaking. In order to prove the effectiveness of resource brokers and associated scheduling algorithms, this paper introduces Economic-based scheduling algorithms in Grid Computing by using a Java-based discrete-event Grid simulation toolkit called GridSim. The toolkit supports modeling and simulation of heterogeneous Grid resource, users and application models.

Key words - Grid Computing; GridSim; Resource; Resource Broker; Processing Elements.

1. Đặt vấn đề

Tính toán lưới (Grid Computing) là một hướng nghiên cứu tính toán mới cho phép chia sẻ, chọn lựa và kết hợp các tài nguyên không đồng nhất phân tán về mặt địa lý với các miền thời gian khác nhau để giải quyết trên quy mô lớn các vấn đề của khoa học, kỹ thuật và quản lý kinh tế.

Quản lý và lập lịch các tài nguyên ứng dụng trên môi trường phân tán với quy mô lớn là một nhiệm vụ phức tạp. Khả năng, cách sử dụng và các điều khoản chi phí khác phụ thuộc vào người sử dụng, thời gian, độ ưu tiên và các mục đích khác nhau. Do đó, việc ứng dụng tính toán lưới với cách tiếp cận phân cấp và không tập trung để quản lý và lập lịch các tài nguyên, nhằm khai thác tối đa các tài nguyên trên môi trường phân tán đang là hướng nghiên cứu hiện nay và đang được nhiều người quan tâm.

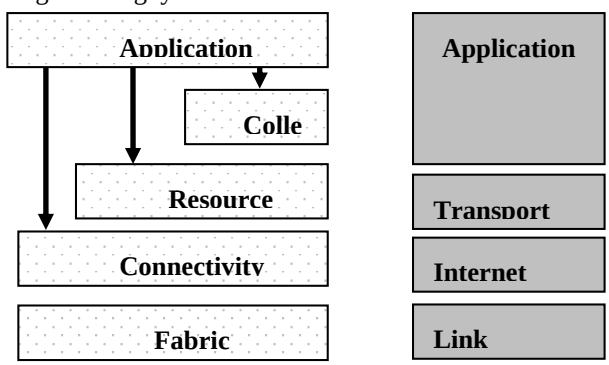
2. Hệ thống tính toán lưới (Grid computing)

Grid computing là bước phát triển tiếp theo của hướng tính toán phân tán, với mục đích cung cấp những dịch vụ tính toán đơn giản cho người dùng, nhưng mang lại sức mạnh tính toán rất lớn bởi tính trong suốt và khả năng kết nối các hệ thống không đồng nhất nhằm chia sẻ các nguồn tài nguyên đa dạng [4]. Có thể xem GridComputing là một loại hệ thống gồm hạ tầng phần cứng và phần mềm phân bố trên mạng cho phép tính toán, lưu trữ và khai thác phân tán, cung cấp cho người dùng với một môi trường chia sẻ tài nguyên cộng tác để giải quyết bài toán có khối lượng lớn.

Một hệ thống Grid Computing bao gồm nhiều nút lưới mà mỗi nút có thể là một lưới nhỏ hay một cluster. Một Grid có cấu trúc phân lớp bao gồm 4 thành phần chính:

Tầng Fabric: Cung cấp các loại tài nguyên chia sẻ được phép truy cập thông qua các giao thức Grid, bao gồm: tài nguyên tính toán, hệ thống lưu trữ, catalogs, tài nguyên mạng và đầu dò (sensors).

Tầng Connectivity: là tầng tạo nên hạt nhân của các giao thức xác thực và truyền thông bắt buộc của các giao dịch trong hệ thống Grid. Giao thức truyền thông cho phép chuyển đổi dữ liệu qua lại giữa các loại tài nguyên ở tầng nền (Fabric). Giao thức xác thực (authentication) được xây dựng trên các dịch vụ truyền thông để cung cấp các cơ chế mã hóa bảo mật trong việc kiểm tra sự xác thực của người dùng và tài nguyên.



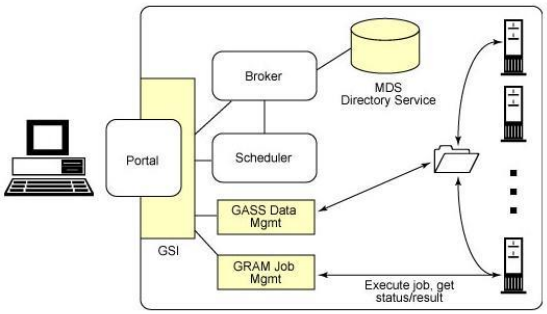
Hình 1. *Kiến trúc phân lớp và các thành phần của Grid.*

Lớp tài nguyên (Resource): được xây dựng trên nền tảng sẵn có của tầng Connectivity. Đây là tầng dùng để xác định các giao thức cho các quá trình thương lượng, khởi tạo, kiểm tra, điều khiển, tính toán và chi phí của các thao tác được chia sẽ trên tài nguyên. Những giao thức trong tầng này sẽ gọi các hàm trong tầng Fabric để truy cập và sử dụng các loại tài nguyên cục bộ.

Tầng Collective: chứa các giao thức và dịch vụ cho phép giao tiếp giữa các tài nguyên trong tập hợp.

Tầng Application: tập hợp các ứng dụng cho phép người dùng có thể phát triển dịch vụ, truy cập và sử dụng tài nguyên trong hệ thống Grid. Tuy nhiên, khi xem xét đến

từng lưới cụ thể nếu nhìn hệ thống từ ngoài vào, bao gồm các thành phần chính như sau:



Hình 2. *Các thành phần của một lưới.*

+ Cổng giao diện với người dùng – Portal
+ Thành phần an ninh – Security (GSI)
+ Thành phần môi giới – Broker
+ Thành phần lập lịch – Scheduler
+ Thành phần quản trị dữ liệu – GASS Data management
+ Thành phần quản trị công việc và tài nguyên – GRAM Job and resource management
+ Dịch vụ thư mục – Meta Directory Service (MDS)

Hình 2 mô tả quá trình người dùng tương tác với lưới thông qua các thành phần này. Người dùng đưa các yêu cầu của mình thông qua portal. Việc kiểm tra xác nhận, ủy quyền cho người dùng sẽ được thành phần an ninh chịu trách nhiệm. Yêu cầu của người dùng sẽ được thành phần môi giới đưa đến bộ lập lịch, cùng với các thông tin do dịch vụ thư mục (MDS – Meta Directory Service) cung cấp. Bộ lập lịch đưa ra quyết định về việc các nhiệm vụ tương ứng trong yêu cầu được thực hiện trên các tài nguyên cụ thể nào, sẽ chuyển những thông tin này cho các thành phần quản trị công việc và dữ liệu để chúng đảm trách việc thực thi công việc cũng như nhận kết quả trả về. Mọi thông tin trả về sẽ được hiển thị trên portal cho người dùng.

3. Vấn đề lập lịch tài nguyên trong tính toán lưới

3.1. Giới thiệu

Lên lịch công việc và cân bằng tải là chức năng quan trọng trong hệ thống tính toán lưới.

Hầu hết hệ thống tính toán lưới bao gồm một số loại hệ thống lịch công việc. Điều này có thể thực hiện bằng cách sử dụng hàng đợi công việc có độ ưu tiên khác nhau. Khi một máy trạm trở nên sẵn sàng thi hành công việc, công việc lấy từ hàng đợi có độ ưu tiên cao nhất sẽ làm trước.

Lịch trình thường tác động đến tình trạng tải của lưới tức thời, dùng thông tin về tình trạng sử dụng hiện tại của máy tính để dò ra các máy rỗi và phân chia công việc xử lý. Lịch trình có thể tổ chức trên một hệ thống phân cấp. Ví dụ một lịch trình cấp lưới (grid) có thể giao việc cho lịch trình cấp bó (cluster) hay các lịch trình cấp thấp hơn chứ không chỉ cho một máy cụ thể.

Các lịch trình tiên tiến sẽ giám sát tiến độ của lịch trình công việc quản lý toàn bộ luồng công việc. Nếu công việc bị mất quyền với hệ thống hay mạng ngắt quãng, một lịch trình tốt sẽ tự động giao lại công việc cho chỗ khác. Tuy nhiên, nếu một công việc rơi vào vòng lặp vô hạn và tiến đến thời gian ngưng trệ cực đại, thì không được làm lại.

Tài nguyên dự trữ trên lưới trong quá trình phát triển được hoàn thành với một hệ thống hạn chế. Nó còn hơn là một lịch trình. Trước hết nó là hệ thống lịch cơ bản để dự trữ tài nguyên cho một khoảng thời gian đặc biệt và chặn các lịch trình khác giành lấy cùng một tài nguyên cùng một thời điểm. Nó cũng có thể gỡ bỏ hay ngừng công việc có thể đang chạy trên bất kì máy hay tài nguyên nào khi đi vào thời ki dự trữ.

3.2. Bài toán lập lịch trong tính toán lưới

Trong tính toán lưới, thành phần lập lịch đóng vai trò lớn trong việc quyết định đến thành công của lưới, vì vấn đề của lưới vẫn là tìm ra tài nguyên thích hợp để thực hiện các nhiệm vụ mà người sử dụng giao phó. Do đặc điểm của lưới là được xây dựng trên cơ sở các hệ thống không đồng nhất và phân tán về mặt địa lý, nên số lượng, chủng loại tài nguyên rất phong phú đa dạng. Vì vậy mức độ phức tạp của bài toán lập lịch rất lớn do phải phụ thuộc vào nhiều yếu tố khách quan.

Khi đưa ra một giải pháp trong thực tế, người phát triển hệ thống phải căn cứ vào tình hình cụ thể của lưới cũng như các yêu cầu tối ưu đặt ra mà quyết định hướng tiếp cận cụ thể cho công việc cài đặt. Việc lựa chọn chính xác đóng vai trò rất quan trọng, vì nó sẽ ảnh hưởng đến hiệu năng, tính tối ưu cũng như tính kinh tế của hệ thống lưới sau này. Trên phương diện nghiên cứu, các nhà nghiên cứu có thể dựa vào các yêu cầu trong thực tế để từ đó đặt ra các mô hình bài toán cho cách tiếp cận của mình. Mỗi cách tiếp cận đều có ưu điểm riêng và những điểm mạnh này cũng đã được chứng tỏ qua các thí nghiệm của họ trên môi trường mô phỏng. Các hướng tiếp cận tiêu biểu hiện đang được nhiều người quan tâm như sau:

3.2.1. Hướng tiếp cận nhằm nâng cao thông lượng của hệ thống

Đó là hướng tổ chức lập lịch cho một tập các nhiệm vụ độc lập với nhau, nhằm nâng cao khả năng xử lý của hệ thống trong một khoảng thời gian dài. Cụ thể hơn: với một tập các nhiệm vụ (meta-task) được gửi đến yêu cầu lập lịch, bộ lập lịch phải sắp xếp sao cho thời điểm kết thúc nhiệm vụ muộn nhất là sớm nhất có thể được.Trong khi đưa ra giải thuật lập lịch theo hướng này, các nhà nghiên cứu đã đề nghị đưa thêm tham số về chất lượng dịch vụ (QoS – Quality of Service), cụ thể trong thí nghiệm là tham số về băng thông của mạng. Các nhiệm vụ có QoS ở mức cao sẽ được đem xét lựa chọn tài nguyên trước các nhiệm vụ có QoS thấp hơn. Trong khi lựa chọn tài nguyên cho một nhiệm vụ, hàm lượng giá về thời điểm hoàn thành tác vụ (CT – Completion Time) sẽ được đánh giá theo heuristic Min–min để quyết định nhiệm vụ nào sẽ được thực hiện trên tài nguyên nào.

3.2.2. Hướng tiếp cận xác lập trước tài nguyên

Một trong những yêu cầu nâng cao cho bộ lập lịch trong tính toán lưới là khả năng xử lý về đặt trước tài nguyên. Trong thực tế, điều này có ý nghĩa rất lớn, nó giúp người sở hữu tài nguyên sẽ tận dụng được tối đa tài nguyên của mình trong việc cho thuê, còn người dùng nếu không có yêu cầu quá cấp bách về thời hạn hoàn thành ứng dụng, có thể lùi thời điểm thực hiện lại với hi vọng về một giá cả chấp nhận được. Trong giải thuật này, tham số QoS chính là khả năng phục vụ của CPU đối với ứng dụng (tải của CPU). Các yêu cầu về tài nguyên sẽ được xử lý theo lô,

theo các khoảng thời gian Δt hoặc sẽ được xử lý ngay khi có yêu cầu ngay lập tức về tài nguyên. Mỗi yêu cầu đều được gán cho một độ ưu tiên và có một hàm lợi ích khi nó được thực hiện trên một tài nguyên nào đó. Tuy nhiên, giải thuật này vẫn còn hạn chế là với mỗi yêu cầu gửi đến chỉ được phép đòi hỏi một tài nguyên.

3.2.3. Hướng tiếp cận mang tính kinh tế

Các hướng tiếp cận trên đều hướng đến các khía cạnh cụ thể của một hệ thống tính toán lưới. Lập lịch các tài nguyên lưới mang tính kinh tế là một hướng tiếp cận khá mới, phản ánh lưới như một thị trường kinh doanh các loại tài nguyên khác nhau. Ý tưởng này chính là nền tảng cho việc xây dựng hệ thống chương trình mô phỏng lập lịch các tài nguyên mang tính kinh tế trong tính toán lưới.

3.3. Vấn đề lập lịch mang tính kinh tế trong tính toán lưới

Khi tính toán lưới phát triển và việc chia sẻ tài nguyên giữa các tổ chức ảo có thêm màu sắc kinh tế, nghĩa là việc sử dụng tài nguyên đồng nghĩa với việc đi thuê và phải trả tiền, thì ta có thể xem xét môi trường lưới như một thị trường kinh doanh, trong đó, các tổ chức ảo có thể đóng vai trò là *người sở hữu tài nguyên* (resource owners) hoặc *người tiêu dùng tài nguyên* (resource consumer) hoặc cả hai. Việc tham gia vào thị trường lưới của những người này đều nhằm mục đích cực đại hoá lợi ích của bản thân. Đối với người sở hữu tài nguyên, họ luôn mong muốn tài nguyên của mình được tận dụng triệt để với giá cho thuê hợp lý trong hoàn cảnh phải cạnh tranh với những nhà cung cấp dịch vụ lưới khác, để cuối cùng thu được lợi nhuận tối đa. Còn đối với người tiêu dùng tài nguyên, họ hi vọng giải quyết được bài toán của mình với chi phí hợp lý và trong một khoảng thời gian hạn định. Việc thoả mãn yêu cầu về hạn định thời gian là rất quan trọng vì đối với một số chuyên ngành như dự báo chẳng hạn, thì sự chậm trễ trong tính toán sẽ làm cho kết quả không còn ý nghĩa thực tế nữa. Bên cạnh đó, có một điều rất quan trọng là: người sở hữu và người tiêu dùng tài nguyên đều phân tán về mặt địa lý, và do đó, đặt ra nhiều thách thức cho việc quản lý các nguồn tài nguyên không đồng nhất và phân tán về mặt địa lý. Với tình trạng như vậy, các hướng tiếp cận mang tính tập trung cao sẽ không còn phù hợp cho môi trường lưới. Người ta cần đến hướng tiếp cận mang tính phi tập trung và một mô hình quản lý tài nguyên cũng như lập lịch mang tính kinh tế.

Những hướng tiếp cận cũ cho bài toán lập lịch trong môi trường lưới quan tâm nhiều hơn đến hệ thống. Việc tính giá dịch vụ cho người dùng là hoàn toàn tĩnh, bộ lập lịch không quan tâm đến thời hạn cần hoàn thành công việc mà chỉ cốt sao lập lịch để tổng thời gian hoàn thành tập công việc là nhỏ nhất có thể. Cũng không có sự thương lượng về giá cả thực hiện ứng dụng giữa người dùng và hệ thống, vì giá này đã được đặt cố định trong hệ thống. Bên cạnh đó, ý tưởng về giá cả thay đổi động theo thời điểm thực hiện ứng dụng cũng khá mới mẻ. Ví dụ như, việc chạy ứng dụng vào ban đêm sẽ rẻ hơn ban ngày chẳng hạn; như thế sẽ khuyến khích những khách hàng ở nửa bên kia bán cầu thuê tài nguyên của chúng ta trong khi chúng ta đang ngon giấc. Những ý tưởng mới trong hướng tiếp cận mang tính kinh tế sẽ giúp người sở hữu tài nguyên kiếm được nhiều tiền hơn từ việc cho thuê tài nguyên

4. Các thuật toán lập lịch

Việc quản lý và lập lịch các tài nguyên trên môi trường lưới, nhằm thỏa mãn tối đa người dùng và thoả mãn tối đa quyền sử dụng tài nguyên. Để làm được điều này, bộ lập lịch (scheduler) có ba vấn đề cần chú ý:

Thứ nhất, scheduler có thể giới hạn tài nguyên sử dụng của mỗi user bằng cách phân quyền sử dụng tài nguyên.

Thứ hai, scheduler có thể lập lịch công việc dựa vào thời điểm kết thúc công việc.

Cuối cùng, scheduler ghi lại và tải các tài nguyên mà mỗi user sử dụng.

Để giới hạn tài nguyên sử dụng, scheduler cần phải phân loại tài nguyên thành nhiều lớp. Trong mỗi lớp có *Hàng đợi* riêng và có các thuộc tính như sau:

Số máy (tài nguyên) trong mỗi lớp và xác định mức lập trình đa nhiệm của máy tương ứng.

User có thể sử dụng tài nguyên nào trong mỗi lớp và số tài nguyên mà mỗi user có thể sử dụng.

Phương pháp lập lịch được sử dụng là tối đa công việc mà hàng đợi có thể lập lịch và tối đa công việc mà hàng đợi có thể lưu trữ.

Để lập lịch công việc dựa vào thời điểm công việc kết thúc, scheduler phải biết thời điểm kết thúc của công việc và ước tính được thời gian thực hiện công việc.

Khi công việc kết thúc, bộ lập lịch làm hóa đơn cho user.

Khả năng khai thác hiệu quả tài nguyên của hệ thống phụ thuộc vào các yếu tố sau:

- Thời gian CPU: dành để thực thi mỗi công việc.
- Cách sử dụng Bộ nhớ: bộ nhớ sử dụng cho mỗi công việc.
- Cách sử dụng Đĩa: không gian Đĩa sử dụng cho mỗi công việc.
- Cách sử dụng mạng: số lượng dữ liệu truyền trên mạng.

Trong đó chi phí về sử dụng đĩa là cố định còn các chi phí khác thì không cố định. Chi phí CPU phụ thuộc vào tốc độ CPU. Nếu CPU tốc độ cao thì chi phí cũng cao. Chi phí của bộ nhớ được phân ra thành nhiều lớp phụ thuộc vào quản trị hệ thống. Chi phí của mạng cố định. Tổng chi phí cho một công việc được tính như sau:

Tổng chi phí = (Thời gian CPU * chi phí của tốc độ máy) + (chi phí sử dụng bộ nhớ) + (chi phí Đĩa) + (Sử dụng mạng*chi phí của mạng) [6]

Ngoài ra còn phụ thuộc vào yêu cầu của người sử dụng tài nguyên Grid: thời gian tối đa (deadline) để hoàn thành công việc và chi phí tối đa (budget) mà người dùng trả cho người sở hữu tài nguyên.

4.1. Chỉ định Deadline và Budget

Các công việc được lập lịch trên Grid thông qua các bộ môi giới của user. Bộ môi giới sử dụng D_{FACTOR} (chỉ deadline) và B_{FACTOR} (chỉ budget) để chỉ ra các giá trị deadline và budget tuyệt đối nhằm thực hiện một kịch bản trước lúc thi hành. Miền giá trị của D_{FACTOR} và B_{FACTOR} trong [0.0..1.0]. Một D_{FACTOR} gần bằng 1 biểu thị sự sẵn sàng của user để thiết lập một Deadline được nới rộng hết mức, deadline này đủ để xử lý các ứng dụng ngay cả khi chỉ có các tài nguyên chậm nhất là sẵn có. Tương tự, một B_{FACTOR} gần bằng 1 biểu thị rằng người dùng sẵn sàng gửi

nhiều tiền như được yêu cầu ngay cả khi chỉ có tài nguyên đắt nhất được sử dụng. Giá trị Deadline tuyệt đối:

Deadline= T_{MIN} +D_{FACTOR} *(T_{MAX} - T_{MIN})

- T_{MIN}: thời gian được yêu cầu để xử lý tất cả các công việc theo hướng song song, cho tài nguyên nhanh nhất với độ ưu tiên cao nhất.
- T_{MAX}: thời gian được yêu cầu để xử lý tất cả các công việc tuần tự, sử dụng tài nguyên chậm nhất.
- Một ứng dụng với D_{FACTOR} <0 sẽ không bao giờ được hoàn thành.
- Một ứng dụng với $D_{FACTOR} \geq 1$ sẽ luôn luôn được hoàn thành miễn là một vài tài nguyên sẵn sàng với chia sẻ người dùng cực tiểu trong suốt Deadline. Giá trị Budget tuyệt đối:

Budget= C_{MIN} +B_{FACTOR} *(C_{MAX} - C_{MIN})

- C_{MIN}: chi phí của việc xử lý tất cả các công việc theo hướng song song trong phạm vi Deadline, cho tài nguyên rẻ nhất có độ ưu tiên cao nhất.
- C_{MAX}: chi phí của việc xử lý tất cả các công việc theo hướng song song trong phạm vi Deadline, đối với tài nguyên đắc nhất có độ ưu tiên cao nhất.
- Một ứng dụng với B_{FACTOR} <0 sẽ không bao giờ được hoàn thành.

Một ứng dụng với $B_{FACTOR} \geq 1$ sẽ luôn luôn được hoàn thành miễn là một vài tài nguyên sẵn sàng chia sẻ với người dùng cực tiểu trong suốt Deadline.

4.2. Thuật toán lập lịch tối ưu chi phí

Thuật toán lập lịch tối ưu chi phí sử dụng các tài nguyên rẻ nhất để đảm bảo Deadline có thể và chi phí tính toán là nhỏ nhất. Thuật toán lập lịch tối ưu chi phí cố gắng để hoàn thành việc thử nghiệm càng tiết kiệm càng tốt trong giới hạn Deadline. Các bước chính của thuật toán như sau:

Đầu vào: Tập tài nguyên Grid; Tập người dùng Grid.

1. Sắp xếp các tài nguyên theo hướng chi phí tăng dần.

2. Đối với mỗi tài nguyên trong sự sắp xếp, phân công càng nhiều công việc càng tốt đến các tài nguyên rẻ nhất, không cần quan tâm đến Deadline.

Đầu ra: Thứ tự các người dùng Grid với chi phí thực hiện công việc theo yêu cầu.

4.3. Thuật toán lập lịch tối ưu thời gian

Thuật toán lập lịch tối ưu thời gian sử dụng tất cả các tài nguyên đủ khả năng để xử lý tất cả các công việc theo hướng song song càng nhanh càng tốt. Các bước chính của thuật toán như sau:

Đầu vào: Tập tài nguyên Grid; Tập người dùng Grid.

1. Đối với mỗi tài nguyên, tính toán thời gian hoàn thành kế tiếp cho một công việc để được gán dựa vào tính toán các công việc được gán trước đó và tốc độ xử lý công việc.

2. Sắp xếp các tài nguyên theo thời gian hoàn thành kế tiếp.

3. Gán một công việc đến tài nguyên đầu tiên mà chi phí đối với công việc này nhỏ hơn hoặc bằng budget còn lại của công việc.

4. Lặp lại các bước trên cho đến khi tất cả các công việc đều được gán.

Đầu ra: Thứ tự các người dùng Grid với thời gian hoàn tất công việc theo yêu cầu.

4.4. Thuật toán lập lịch tối ưu chi phí và thời gian

Thuật toán lập lịch tối ưu chi phí và thời gian tương tự thuật toán tối ưu chi phí, nhưng nếu nhiều tài nguyên cùng chi phí, áp dụng thuật toán tối ưu thời gian. Các bước chính của thuật toán như sau :

Đầu vào: Tập tài nguyên Grid; Tập người dùng Grid.

1. Khám phá tài nguyên: Dựa vào GridInformation Service xác định các tài nguyên và khả năng của chúng.

2. Trao đổi tài nguyên: Xác định chi phí của tất cả các tài nguyên và khả năng được phân phối trên mỗi đơn vị chi phí.

3. Lập lịch: Lặp trong khi hiện tại còn tồn tại các công việc chưa được xử lý và các chi phí xử lý trong phạm vi giới hạn Deadline và Budget.

4. Phân phối: Xác định số lượng các công việc có thể được đệ trình mà không quá tải tài nguyên. Phương thức mặc định là sẽ gửi các công việc miễn là số các công việc của người dùng được triển khai (hoạt động hay trong hàng đợi) ít hơn số các PE trong tài nguyên.

Đầu ra: Thứ tự các người dùng Grid với chi phí tối đa mà người dùng phải trả để hoàn tất công việc theo yêu cầu nhanh nhất.

5. Xây dựng hệ thống mô phỏng lập lịch các tài nguyên

Để xây dựng hệ thống chương trình mô phỏng lập lịch các tài nguyên trên Grid, sử dụng bộ công cụ GridSim, chúng ta thực hiện các công việc như sau:

1. Xây dựng mô hình mô phỏng môi trường Grid Computing.

2. Mô phỏng thuật toán lập lịch tối ưu thời gian.

3. Mô phỏng thuật toán lập lịch tối ưu chi phí.

4. Mô phỏng thuật toán lập lịch tối ưu thời gian và chi phí.

Chương trình xây dựng các module độc lập để thực hiện các công việc trên.

5.1. Xây dựng mô hình mô phỏng môi trường Grid Computing

Mô hình hoá môi trường Grid gồm các bước sau:

1. Khởi tạo các tham số ban đầu để khởi động gói GridSim.

2. Mô hình hoá tài nguyên Grid (Resource).

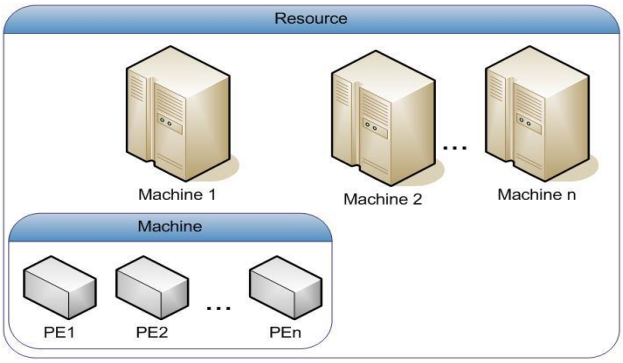
3. Mô hình hoá người dùng Grid (User).

4. Thực thi quá trình mô phỏng.

5.1.1. Mô hình hoá tài nguyên Grid

Mô hình tài nguyên Grid được chỉ ra trong hình 4.1. Trong mô hình này ta có thể tạo ra các phần tử xử lý (PE-Processing Element) với các tốc độ khác nhau. PE là đơn vị nhỏ nhất, và trong thực tế, ta có thể coi mỗi PE tương ứng với 1 CPU, thực hiện các nhiệm vụ tính toán. Sau đó, một hoặc nhiều PE có thể được đặt cùng nhau để tạo thành một máy (Machine). Tương tự, một hoặc nhiều máy có thể được đặt cùng nhau để tạo ra một tài nguyên Grid (Grid Resource). Tài nguyên Grid kết quả có thể là một bộ xử lý đơn, đa xử lý chia sẻ bộ nhớ (SMP-Shared memory multiprocessor), hoặc một nhóm bộ nhớ phân tán của các máy tính, trong thực tế nó có thể tương đương với một cluster. Các tài nguyên Grid này có thể mô phỏng lập lịch chia sẻ thời gian hoặc bộ nhớ phụ thuộc vào các phương thức phân phối. Một PE đơn hoặc SMP kiểu tài nguyên

Grid được quản lý tiêu biểu bởi các hệ điều hành chia sẻ thời gian sử dụng phương thức lập lịch Round-Robin. Các hệ thống đa xử lý bộ nhớ phân tán được quản lý bởi các hệ thống hàng đợi, được gọi là các bộ lập lịch chia sẻ bộ nhớ, thực thi một công việc trên một PE dành riêng khi được cấp phát. Với cách thức mô hình hoá này thể hiện tính mềm dẻo vào đảm bảo khả năng mô hình hoá nhiều kiểu tài nguyên khác nhau trong thực tế.



***Hình 3**. Mô hình hoá tài nguyên Grid.*

Trong đó mỗi máy gồm danh sách các PE: Id của PE, tốc độ MIPs (Triệu lệnh trên một giây). Số các máy trong một Resource sẽ được lấy ngẫu nhiên. Số các PE của một máy cũng được lấy ngẫu nhiên.

Các phương thức chính của lớp addResource:

public addResource(int id, string name, Random r)

public String getGridName()

public int getGridId()

public void setGridName(String name)

public double getBaudRate()

public int getPeakLoad()

public int getOffLoad()

public int getHolidayLoad()

public ResourceCharacteristics getResourceCharacteristics()

public void setBaudRate(double baudRate)

public void setPeakLoad(int peakLoad)

public void setOffLoad(int OffLoad)

public void setHolidayLoad(int holidayLoad)

public Vector getListMachineName()

public void defaultResource()

5.1.2. Mô hình hoá người dùng Grid

Tạo ra một hoặc nhiều User sử dng tài nguyên Grid. Mỗi User gồm nhiều Gridlet được lưu trong danh sách listgridlet. Số các Gridlet cho mỗi User được lấy ngẫu nhiên. Mỗi Gridlet của User gồm các thông tin như: số thứ tự, chiều dài, kích thước file đầu vào, kích thước file đầu ra. Như vậy, ta có thể thực hiện vòng lặp để tạo các thông số ngẫu nhiên cho Gridlet. Mỗi User chứa đựng trong đó phương thức body() mô tả quá trình hoạt động của User.

Các phương thức chính của lớp addUser bao gồm:

public java.util.Vector createUser(int num, boolean random, int total_resource)

public void removeAllUser()

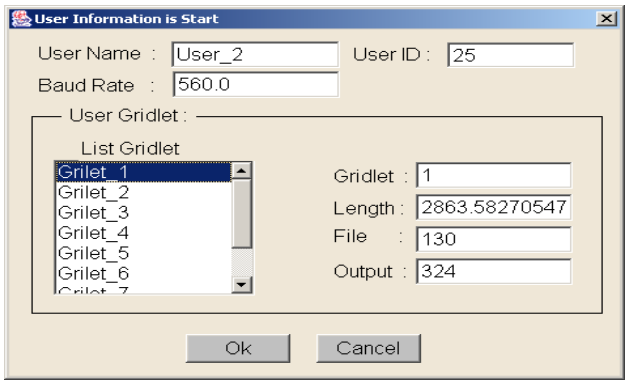
public int removeUser(String value)

public void body ()

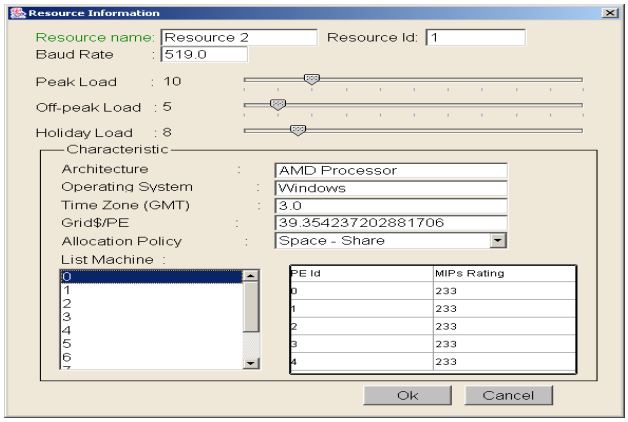
public int getTotalUser()

public int getUserCounter()

Chương trình cung cấp giao diện cho phép mô phỏng xây dựng môi trường GridComputing như: tạo mới, thêm, xoá các User, các Resource: Quản lý các thông tin của mỗi Grid User; Quản lý các thông tin của mỗi Grid Resource.



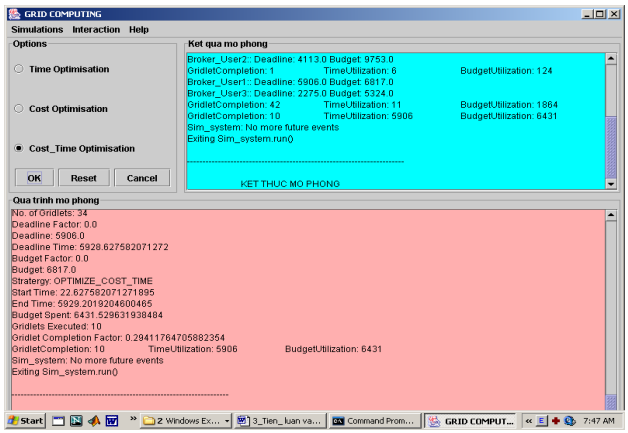
***Hình 4**. Quản lý thông tin mỗi Grid User.*



***Hình 5.** Quản lý thông tin mỗi Grid Resource.*

5.2. Mô phỏng thuật toán lập lịch tối ưu chi phí và thời gian

Theo chiến lược lập lịch tối ưu chi phí và thời gian, bộ lập lịch sẽ lấy chi phí làm tiêu chí. Nếu User đưa ra chi phí phải trả cao, sẽ được lập lịch trước. Nếu các User đưa ra cùng chi phí thì sẽ dựa vào thời gian yêu cầu hoàn thành công việc, ai có yêu cầu thời gian hoàn thành công việc thấp sẽ được lập lịch trước.



***Hình 6**. Kết quả mô phỏng lập lịch tối ưu chi phí và thời gian*

Như vậy, qua thực nghiệm dựa trên môi trường Grid Computing đã tạo ra ở trên. Kết quả mô phỏng thuật toán lập lịch tối ưu thời gian và chi phí như sau: Đầu tiên, bộ lập lịch sẽ lập lịch cho User2 được sử dụng tài nguyên Grid. Số các Gridlet của User2 thành công là 1. Chi phí mà User2 phải trả cho người sở hữu tài nguyên là 9753.0 (đvtt). Tiếp đến, lập lịch cho User1. Số các Gridlet của User1 thành công là 34. Chi phí mà User1 phải trả cho người sở hữu tài nguyên là 6817.0 (đvtt). Cuối cùng, User3 được lập lịch. Số các Gridlet của User3 thành công là 42. Chi phí mà User3 phải trả cho người sở hữu tài nguyên là 5324.0 (đvtt).

Qua một số các thử nghiệm với các giá trị Deadline và Budget thay đổi. Ta có thể rút ra được các đánh giá sau:

Với thuật toán lập lịch tối ưu thời gian : Số Gridlet được xử lý tăng lên nếu giá trị budget và deadline tăng. Nghĩa là khi giá trị budget cao hơn, bộ môi giới sẵn sàng chọn tài nguyên đắt để xử lý được nhiều công việc, dẫn đến thời gian hoàn thành công việc giảm.

Với thuật toán lập lịch tối ưu chi phí : Deadline không lớn, muốn số gridlet được xử lý tăng lên thì giá trị budget phải tăng. Do khi budget cao hơn, broker sẵn sàng thuê tài nguyên để xử lý được nhiều công việc trong phạm vi deadline. Hoặc khi lập lịch với giá trị budget giảm, muốn số các gridlet được xử lý tăng lên thì deadline phải được giãn ra.

6. Kết luận

Dựa trên việc tìm hiểu và nghiên cứu các thuật toán lập lịch trên hệ thống GridComputing là cơ sở để có thể triển khai xây dựng hệ thống GridComputing tại phòng thí nghiệm mạng của Khoa CNTT trường Đại học Bách Khoa nhằm phát triển các hệ thống tính toán khối lượng lớn.

TÀI LIỆU THAM KHẢO

[1] Trần Hồ Thủy Tiên, *Lập lịch các tài nguyên trong tính toán lưới*, Luận văn Thạc sĩ chuyên ngành Công nghệ thông tin, Khóa 2001-2004.

[2] M. Miller and K. Drexler, *Markets and Computation: Agoric Open Systems*, The Ecology of Computation, B. Huberman (editor), Elsevier Science Publishers, The Netherlands, 1998.

[3] M. Neary, A. Phipps, S. Richman, P. Cappello, *Javelin 2.0: Java-Based Parallel Computing on the Internet*, Proceedings of European Parallel Computing Conference (Euro-Par 2000), Germany, 2000.

[4] R. Buyya, High Performance Cluster Computing: Architectures and Systems, Volumes 1 and 2, Prentice Hall, NJ, USA, 1999.

[5] R. Buyya and M. Murshed, GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing, Technical Report, Monash University, 2001

[6] R. Buyya et. al., The HEPGrid (High Energy Physics and the Grid Network) Project, http://www.gridbus.org/hepgrid/

[7] R. Raman and M. Livny, Matchmaking: Distributed Resource Management for High ThroughputComputing, Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, July 28-31, 1998, Chicago, IL.

(BBT nhận bài: 28/07/2015, phản biện xong: 28/10/2015)