Có sẵn trực tuyến tại www.sciencedirect.com

Procedia Computer Science 4 (2011) 166–175

Hội nghị Quốc tế về Khoa học Tính toán, ICCS 2011

Framework tối ưu hóa đa mục tiêu động để lựa chọn

các Triển khai phân tán trong môi trường không đồng nhất

Elisabeth Vineka,1, Peter Paul Beranb,, Erich Schikutab

*aCERN, CH-1211 Gene`ve 23, Switzerland*

*bĐại học Vienna, Nhóm Công nghệ và Hệ thống Quy trình Làm việc, Rathausstr. 19 / 9, A-1010 Vienna, Áo*

**Tóm tắt**

Trong các hệ thống phân tán, nơi tồn tại một số triển khai của một dịch vụ cụ thể, nhiệm vụ quan trọng là phải chọn và kết hợp các triển khai cụ thể để xây dựng một qui trình làm việc có thể thực thi, các thuộc tính phi chức năng như hiệu suất và tính sẵn có được tính đến trong các qui trình lựa chọn, được thiết kế để đạt được các mục tiêu nhất định trong khi đáp ứng các ràng buộc.Trong bài báo này, một kịch bản cụ thể ứng dụng chuyên sâu về dữ liệu từ một thí nghiệm vật lí năng lượng phóng xạ cao bao gồm một thử thách lựa chọn triển khai được đưa ra. Một mô hình chung cho các hệ thống phần tán được chỉ định dựa trên sự bắt nguồn của một mô hình chính thức đại diện cho các thành phần riêng lẻ của hệ thống. Vấn đề tối ưu hóa được tiếp cận cả từ góc độ của người dùng và góc độ của nhà sản xuất hệ thống. Hơn nữa nhưng khía cạnh động của hệ thống cơ bản cũng được tính đến, điều này dẫn đến một vấn đề tối ưu hóa theo đa mục tiêu động, từ đó đề xuất một thuật toán di truyền dựa trên bộ nhớ rõ ràng.

*Từ khóa*: Lựa chọn dịch vụ, Tối ưu hóa đa mục tiêu, Thuật toán di truyền.

# Giới thiệu

Với sự ra đời của các hệ thống phần tán – có thể là môi trường lưới, môi trường đám mây hoặc môi trường dịch vụ web. Việc lựa chọn tài nguyễn để thực hiện một công việc nhất định, tưởng ứng với việc lựa chọn một dịch vụ dể đáp ứng một yêu cầu nhất định, đã trở thành một nhiệm vụ quan trọng. Việc các hê thống phần tán đã xuất hiện trong vài thập kỷ nay, tạo ra nhiều cách tiếp cận đối với thử thách lựa chọn tài nguyễn hay dịch vụ. Chúng đều nhằm mục đích cho phép một số tối ưu hóa, tùy thuộc vào yêu cầu của hệ thống. Giảm thiểu thời gian thực hiện công việc, giảm thiểu chi phí liên quan đến một công việc, tối đa hóa thông lượng toàn hệ thống và tối đa hóa một số chức năng tiện ích được xác định tùy chỉnh là những ví dụ về sự tối ưu hóa nói trên. Trong một hệ thống mà một số triển khai của một chức năng xác đinh (dịch vụ) tồn tại quyết định về việc thực thi những chức năng nào có thể dựa trên sự đánh giá các thuộc tính phi chức năng, chẳng hạn như các chỉ số hiệu suất, tính sẵn có và độ tin cậy, và trạng thái hiện tại của triển khai tại thời điểm yêu cầu.

Trong bài báo này, chúng tôi nghiên cứu thách thức lựa chọn triển khai từ hai góc độ khác nhau và thường mâu thuẫn nhau, cụ thể là từ quan điểm của người dùng và nhà cung cấp hệ thống. Người dùng muốn tối ưu hóa việc thực

*Email addresses:* [elisabeth.vinek@cern.ch](mailto:elisabeth.vinek@cern.ch) (Elisabeth Vinek), [peter.beran@univie.ac.at](mailto:peter.beran@univie.ac.at) (Peter Paul Beran),

[erich.schikuta@univie.ac.at](mailto:erich.schikuta@univie.ac.at) (Erich Schikuta)

1Corresponding author

1877–0509 © 2011 Published by Elsevier Ltd. Open access under [CC BY-NC-ND license.](http://creativecommons.org/licenses/by-nc-nd/3.0/)

Selection and/or peer-review under responsibility of Prof. Mitsuhisa Sato and Prof. Satoshi Matsuoka doi:10.1016/j.procs.2011.04.018

thi các yêu cầu cụ thể của họ (ví dụ: về việc giảm thiểu thời gian thực hiện) mà không cần lo lắng về hậu quả đối với tổng thể hệ thống. Tuy nhiên, mục tiêu của nhà cung cấp là tối ưu hóa thông lượng hệ thống và cho phép sử dụng hợp lý các tài nguyên, hoặc một chế độ sử dụng được quyết định sẵn. Mặc dù người dùng có nhiều khả năng theo đuổi cùng một chiến lược cho mỗi yêu cầu, hệ thống chịu trách nhiệm có thể phải đối mặt với một môi trường luôn vận động và thay đổi, bao gồm thay đổi yêu cầu, thay đổi cách sử dụng và thay đổi các quyết định về mục tiêu kinh doanh. Để giải quyết vấn đề này, chúng tôi đề xuất một framework tối ưu hóa đa mục tiêu để lựa chọn các triển khai phân tán trong một môi trường không đồng nhất. Mô hình của chúng tôi phản ánh sự thay đổi liên tục mà một hệ thống phân tán có thể phải đối mặt.

Ý tưởng này xuất phát từ và dựa trên một bối cảnh ứng dụng vào cuộc sống từ thử nghiệm ATLAS với mục đích khám phá vật lý mới tại Máy gia tốc hạt lớn [1]. Hệ thống TAG [2] là một hệ thống phân phối gồm các cơ sở dữ liệu và một số dịch vụ web có thể truy cập chúng. Mục tiêu chính của nó là cho phép truy vấn dữ liệu nhanh chóng và hiệu quả về các sự kiện đã xảy ra trong máy dò ATLAS và do đó có nghĩa là để tăng tốc quá trình phân tích sử dụng nhiều tài nguyên bằng cách chọn trước dữ liệu có tính hữu dụng. Hệ thống TAG được đặc trưng bởi các thành phần được liên kết chặt chẽ với một bộ điều khiển trung tâm ví dụ và có một môi trường tương đối ổn định và khả năng thu thập kiến ​​thức chuyên sâu về hành vi của các thành phần. Với những đặc điểm này, chúng tôi đã phát triển một đại diện chung của các hệ thống phân tán nói chung và môi trường TAG nói riêng, một mô hình chính thức mô tả các thành phần và dẫn đến vấn đề tối ưu hóa đa mục tiêu và chúng tôi giới thiệu một khung tối ưu hóa đa mục tiêu động dựa trên thuật toán di truyền.

Phần còn lại của tài liệu này được sắp xếp như sau. Phần 2 trình bày chi tiết về kịch bản tạo động lực. Trong phần 3 các khối xây dựng của một hệ thống phân tán nói chung và của hệ thống ứng dụng của chúng tôi nói riêng, được mô tả và đặt trong mối quan hệ. Dựa trên các khối xây dựng này, một mô hình chính thức để mô tả hệ thống TAG được phát triển trong Phần 4. Phần 5 mô tả vấn đề tối ưu hóa đa mục tiêu cụ thể gặp phải trong kịch bản ứng dụng và giới thiệu một cách tiếp cận thuật toán di truyền động để giải quyết nó. Phần 6 thảo luận về công việc liên quan trong lĩnh vực lựa chọn dịch vụ trong các bài toán tối ưu hóa đa mục tiêu. Phần 7 kết luận và nêu chi tiết các bước tiếp theo.

# Kịch bản tạo động lực

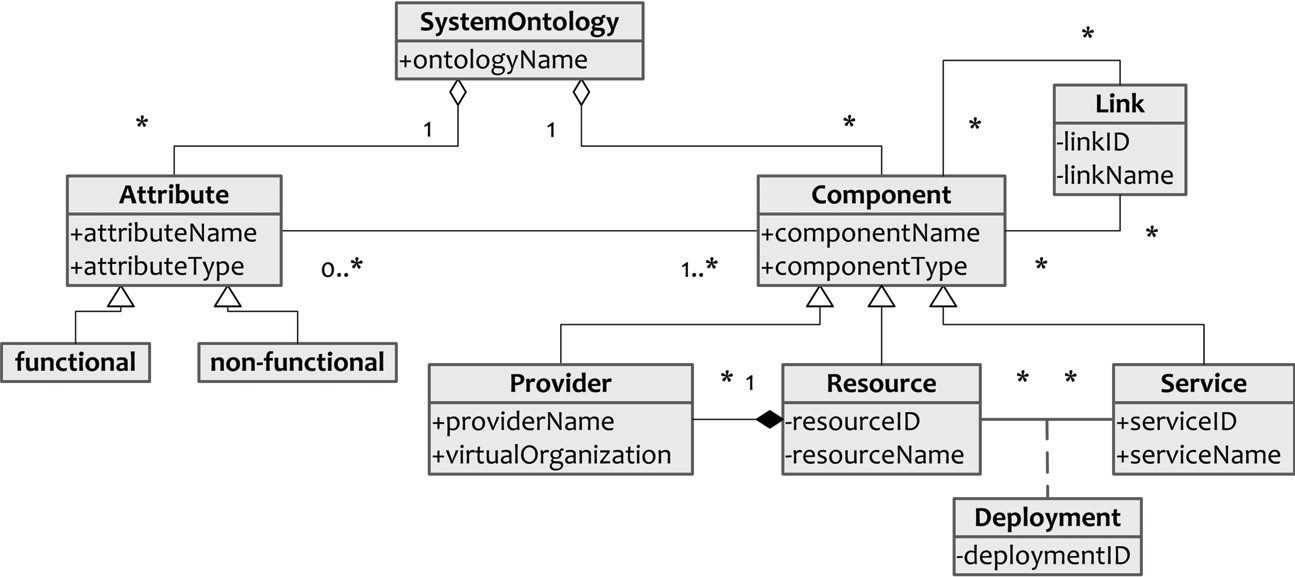
ATLAS là một trong những máy dò của Máy va chạm Hadron Lớn (LHC) được tổ chức tại CERN, Tổ chức Nghiên cứu Hạt nhân Châu Âu. Các ATLAS mục đích thí nghiệm tại khám phá vật lý mới bằng cách phân tích một số lượng lớn các dữ liệu từ các va chạm proton-proton [1]. Để cho phép lựa chọn nhanh các sự kiện vật lý mà không cần phải thực sự xem xét bên trong các tập dữ liệu lớn, siêu dữ liệu cấp sự kiện - được gọi là TAG - được tạo ra. Một TAG là một nhỏ, cao cấp tóm tắt của một sự kiện với số lượng vật lý chủ chốt và thông tin cho back-chuyển hướng cho sự kiện cơ bản. TAG được lưu trữ trong cơ sở dữ liệu quan hệ (và trong phạm vi công việc này, nhưng điều này không được xem xét trong phạm vi của công việc này) và chủ yếu là các dịch vụ dựa trên web được cung cấp để truy cập TAG, thực hiện truy vấn, chuyển đổi định dạng đầu ra, chuyển đầu ra cho người yêu cầu và cuối cùng gửi các công việc phân tích lên Grid. Cả cơ sở dữ liệu TAG và các dịch vụ liên quan của chúng đều được phân phối trên khắp thế giới. Hiện tại, một số trang web ở Châu Âu và Canada đang lưu trữ dữ liệu TAG hoặc các dịch vụ, hoặc cả hai. Do đó, một chức năng cụ thể - hoặc dịch vụ - tồn tại ở một số địa điểm. Có một phiên bản điều khiển trung tâm quản lý việc triển khai các dịch vụ tại tất cả các trang web tham gia . Do đó, như phản đối để hầu hết các tình huống lựa chọn dịch vụ được mô tả trong tài liệu, hệ thống được coi là rất nổi tiếng, có thể được theo dõi tích cực và tương đối ổn định. Các trang web và triển khai được thêm vào khi hệ thống phát triển, nhưng ở một tốc độ hợp lý có thể được theo dõi. Việc triển khai của một chức năng đưa ra là thường giống hệt nhau về phiên bản phần mềm của họ (tức là chúng ta không phải là trong sự hiện diện của nhiều di ff erent triển khai của một chức năng, nhưng di ff erent triển khai của cùng một mảnh của phần mềm), nhưng khi họ đang chạy trên máy không đồng nhất tại di ff erent các trang web, họ di ff er trong chức năng phi họ thuộc tính như vậy là hiệu suất và tính sẵn sàng. Khi một người sử dụng bắt đầu một yêu cầu, đó là nhiều khả năng của cơ sở dữ liệu và các dịch vụ kết hợp, một số trong số đó là e ffi cient, trong khi những người khác có thể su ff er từ độ trễ cao, thông lượng mạng kém, hoặc máy bị quá tải. Không chỉ các nguồn cụ thể được sử dụng có những hạn chế của chúng mà còn phải tính đến mối liên hệ giữa chúng. Do đó, đó là một thách thức để lựa chọn và kết hợp cơ sở dữ liệu cụ thể và triển khai dịch vụ khi một yêu cầu được bắt đầu. Đây vấn đề bản đồ để một đa chiều Multi-lựa chọn Knapsack Vấn đề (MMKP) [3] được biết đến là NP-khó theo nghĩa mạnh [4]. Để cho phép một sự lựa chọn như vậy, hệ thống ban đầu phải được ghi nhận một cách đúng đắn và các thành phần riêng lẻ đăng ký theo một cách mà có thể dễ dàng truy vấn. Ngoài ra, các thành phần cần được mô tả theo các thuộc tính phi chức năng của chúng , cách sử dụng hệ thống cần phải được hiểu để lấy ra các mẫu truy vấn và quy trình, và một vấn đề tối ưu hóa phải được xây dựng. Tóm lại, các yêu cầu và mục tiêu sau được giải quyết:

1. Làm cho tất cả cơ sở dữ liệu TAG giống như một, tức là ẩn bất kỳ chi tiết phân phối dữ liệu nào với người dùng.
2. Làm cho việc triển khai dịch vụ trở nên minh bạch với người dùng.
3. Đảm bảo sử dụng hiệu quả *tất cả các*nguồn lực sẵn có từ cả người dùng và hệ thống quan điểm.
4. Cho phép cân bằng tải và không thực hiện được các cơ chế cho tài nguyên và dịch vụ TAG.

Mục tiêu đầu tiên đã đạt được bằng cách triển khai sổ đăng ký siêu dữ liệu không thuộc chủ đề của bài báo này. Các cách tiếp cận cho ba mục tiêu khác được mô tả trong phần còn lại của bài báo này. Bản thể luận hệ thống được trình bày trong phần tiếp theo đã được phát triển để xây dựng mô hình hệ thống TAG phân tán dựa trên cơ sở đó một đăng ký dịch vụ đặc biệt đã được thực hiện. Để giải quyết các mục tiêu 3 và 4, trình tối ưu hóa lựa chọn dịch vụ phải được triển khai dựa trên các mục tiêu tối ưu hóa cụ thể.

# Các thành phần cảu Hệ thống phân tán của chúng tôi

Là bước đầu tiên hướng tới việc phân tích hệ thống TAG phân tán, chúng tôi đã phát triển một mô tả chung về các khối xây dựng của một hệ thống phân tán và các tương tác của chúng. Điều này đã được trình bày trong công trình trước đây của chúng tôi [5]. Hình 1 cho thấy một Sơ đồ lớp UML đại diện cho các phần riêng lẻ của một hệ thống phân tán.



Hình 1: Bản thể học hệ thống (Sơ đồ lớp UML)

Các *hệ thống Ontology*được cấu tạo của *các thành phần*và *Thuộc tính*mô tả chúng. Một thành phần có thể là một trong các lớp sau. Một *nhà cung cấp*là một ảo Tổ chức hoặc chỉ đơn giản là một địa lý trang web mà tham gia vào các hệ thống phân phối được coi là bằng cách lưu trữ ít nhất một tài nguyên. Một *tài nguyên*là một thực thể vật lý hay logic có khả năng lưu trữ một dịch vụ. Ví dụ về tài nguyên là máy chủ web, cơ sở dữ liệu và máy ảo. Tài nguyên thuộc về một nhà cung cấp, như được biểu thị bằng ký hiệu thành phần trong Hình 1. Một *Dịch vụ*là một chức năng chung, được mô tả bằng các thuộc tính chức năng. Sự kết hợp của các dịch vụ của hệ thống hoàn toàn mô tả chức năng của nó. Một *triển khai*là phần gọi bởi một khách hàng. Nó được mô hình hóa như một lớp liên kết giữa *Tài nguyên*và *Dịch vụ*, tức là nó là một thể hiện cụ thể của một dịch vụ chung, chạy trên một tài nguyên cụ thể.

Ngoài ra, một lớp *Liên kết*được xác định giữa các thành phần. Khi một số triển khai được kết hợp để tạo thành một công việc phức tạp, liên kết giữa chúng - thường đề cập đến liên kết mạng - là điều quan trọng cần xem xét, vì việc truyền dữ liệu có ảnh hưởng đến hiệu suất tổng thể. Trong các giới hệ thống ontology, tuy các liên kết được xác dịnh ở cấp độ thành phần nhưng ta có thể duy trì tính khả dụng liên quan đến định nghĩa của nó trong mô tả hệ thống cụ thể. Trong một số hệ thống, có thể xác định và mô tả mối liên kết giữa các triển khai, nhưng trong một số hệ thống khác, có thể xác định được liên kết giữa các nhà cung cấp hoặc tài nguyên.

Mỗi thành phần được mô tả bởi một hoặc một số thuộc tính phân biệt. Nói chung, các thuộc tính có thể được phân loại thành các thuộc tính có chức năng và phi chức năng. Chức năng thuộc tính mô tả các chức năng nhất định của một thành phần, thường bao gồm cả nguyên liệu đầu vào, hành vi và kết quả đầu ra. Do đó, chúng mô tả *những gì mà*thành phần cụ thể phải hoàn thành, trái ngược với *cách thức*. Ví dụ về một thuộc tính chức năng là các mẫu truy cập và các chi tiết của logic nghiệp vụ. Thuộc tính phi chức năng hoặc QoS mô tả đặc điểm liên quan đến hoạt động của các thành phần, quy định cụ thể *như thế nào*là một thành phần làm tròn một cho chức năng. Ví dụ cho QoS thuộc tính bao gồm hiệu suất số liệu khác nhau, tính sẵn sàng và độ tin cậy.

Mô hình này là đại diện hệ thống đơn giản hóa, có chứa tất cả các thành phần cần thiết cho chúng tôi dựa trên kinh nghiệm phương pháp tối ưu hóa lựa chọn. Các mô hình rộng hơn đã được đề xuất, chẳng hạn như Mô hình Thông tin Chung (CIM), một tiêu chuẩn DMTF [6]. Các CIM chi tiết tất cả các thành phần hệ thống bao gồm các số liệu và cung cấp phương tiện để hệ thống ghi chép, cơ sở dữ liệu, thiết bị, mạng lưới, sự kiện… vv. Chúng tôi tự phát triển mô hình khái quát đơn giản hóa để hiển thị phạm vi áp dụng của phương pháp của chúng tôi mà không đòi hỏi một hệ thống quá chi tiết đại diện.

Dựa trên mô hình chung đã trình bày, *Danh mục dịch vụ TAG*đã được triển khai dưới dạng lược đồ cơ sở dữ liệu. Tất cả các trang web, tài nguyên, dịch vụ và triển khai đều được đăng ký trong danh mục này và được cập nhật khi có những thay đổi. Bên cạnh đó, triển khai hành đang tích cực đăng nhập và các kết quả thô thống kê được thường xuyên tổng hợp để tính toán ở cấp cao hơn các thuộc tính như tính sẵn sàng và nâng chỉ số đặc trưng này.

# Trình bày chính thức của các thành phần và thuộc tính TAG

Tất cả các phần cần thiết để phác thảo hệ thống phân tán đã trình bày và để thực hiện tối ưu hóa được mô tả trong phần sau. Các phác thảo chính thức mô hình được dựa trên các hệ thống ontology được trình bày trong Mục 3 và sau các phương pháp phát triển trong [7].

* 1. *Thuộc tính*

Một thuộc tính - hoặc chất lượng - được định nghĩa là: *qi* = *id*(*qi*), *Di*, *normi*(*ai*) trong đó *id*(*qi*) là một hàm liên kết một định danh duy nhất với thuộc tính, *Di* là *định nghĩa Miền*và *normi*(*ai*) là một hàm chuẩn hóa chỉ định cách ánh xạ giá trị thuộc tính *ai* thành một giá trị chuẩn hóa trong khoảng [0, 1]. Việc liên kết một định danh duy nhất với từng thuộc tính cho phép làm cho các thuộc tính có thể so sánh được giữa các ranh giới của nhà cung cấp. Tuy nhiên, vì nó không cần thiết để phát triển thêm mô hình chính thức, nó có thể được bỏ qua ngay bây giờ. Do đó, biểu diễn thuộc tính đơn giản là: *qi* = *Di*, *normi*(*ai*). *ai* ∈ *Di* là một giá trị cụ thể mà thuộc tính có thể nhận. Bảng 1 liệt kê hai thuộc tính ví dụ, đó là tính khả dụng và độ trễ.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thuộc tính | Thành phần | *Di* | Giá trị mẫu (*a0i*) | Giá trị chuẩn hóa *normi*(*a0i*) |
| Khả dụng | Triển khai | [0,1] | 0.988 (98.8%) | *normi*(*a*0*i*) ∈ [0, 1] |
| Độ trễ | Liên kết | R+ | 2 ms | *normi*(*a*0*i*) ∈ [0, 1] |

Bảng 1: Ví dụ về thuộc tính

Bản đồ chuẩn hóa để *tăng*giá trị thuộc tính (càng cao càng tốt), chẳng hạn như tính khả dụng được xác định là:



Bản đồ chuẩn hóa để *giảm*giá trị thuộc tính (càng thấp càng tốt), chẳng hạn như độ trễ được xác định là:



* 1. *Các thành phần*

Tiếp theo, chúng tôi chính thức xác định tất cả các thành phần hệ thống. Số thuộc tính khác nhau cho mỗi thành phần được định nghĩa là *k*.

**Nhà cung cấp:** Chúng tôi định nghĩa *P* = *P1* ∪ ... ∪ *Pn* là liên hiệp của tất cả các nhà cung cấp hoặc trang web, trong đó n = P là số lượng nhà cung cấp trong hệ thống của chúng tôi.

Một nhà cung cấp cụ thể Pi, i ∈ {1, ..., n} được xác định là:

P*i* = {*id*(*Pi*), [*qi1*, ..., *qik*]}(1)

**Nguồn lực:** Chúng tôi định nghĩa *R* = *R1*∪ …∪ *Rn* là liên hiệp của tất cả tài nguyên, trong đó n=R là số tài nguyên trong hệ thống của chúng tôi.

Một tài nguyên cụ thể *Ri*, *i* ∈ {1, ..., *n*} được định nghĩa là:

*Ri* = {*id*(*Ri*), *Pj*, [*qi*1, ..., *qik*]} (2)

Nói cách khác, phần thuộc về một nhà cung cấp là một phần của định nghĩa tài nguyên. Theo Công thức 1, điều này tương đương với:

*Ri* = {*id*(*Ri*), {*id*(*Pj*), [*qj*1, ..., *q jk*]}, [*qi*1, ..., *qik*]} (3)

**Dịch vụ:** Chúng ta định nghĩa *S = S1* ∪ *...* ∪ *Sn* là liên hợp của tất cả các dịch vụ, trong đó *n = |S|* là số lượng dịch vụ trong hệ thống của chúng tôi.

Một dịch vụ cụ thể S i, i ∈ {1, ..., n} được xác định là:

*S i* = {*id*(*S i*), [*qi*1, ..., *qik*]} (4)

**Triển khai:** Chúng ta định nghĩa *D = D1* ∪*…*∪ *Dn* như sự hợp nhất của tất cả các triển khai, trong đó *n = |D|* là số triển khai trong hệ thống của chúng tôi.

Triển khai cụ thể Di, i ∈ {1, ..., n} được xác định là:

*Di* = {*id*(*Di*), *Rj*, *S l*, [*qi*1, ..., *qik*]} (5)

Nói cách khác, tài nguyên và dịch vụ mà một triển khai được liên kết là một phần của định nghĩa triển khai. Theo Công thức 1, 2 và 7, điều này tương đương với:

*Di* = {*id*(*Di*),

{*id*(*Rj*), {*id*(*Pm*), [*qm*1, ..., *qmk*]}, [*qj*1, ..., *q jk*]},

{*id*(*S l*), [*ql*1, ..., *qlk*]},

[*qi*1, ..., *qik*]} (6)

Nói cách khác, định nghĩa của nhà cung cấp, tài nguyên và dịch vụ là một phần của định nghĩa triển khai. Do đó, việc lựa chọn một triển khai cụ thể có nghĩa là chọn một nhà cung cấp, nguồn lực và dịch vụ cụ thể.

**Liên kết:** Chúng tôi định nghĩa *L* = *L*1 ∪ ... ∪ *Ln* là hợp của tất cả các liên kết, trong đó *n* = |*L*| là số lượng liên kết trong hệ thống của chúng tôi.

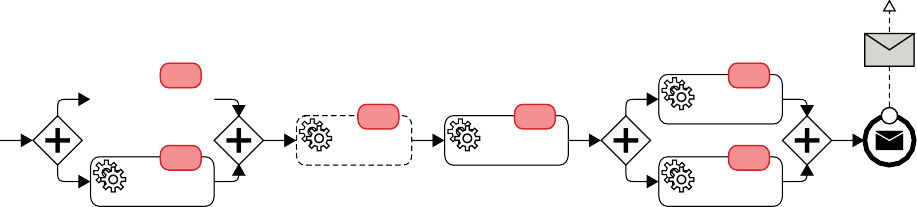
Một liên kết cụ thể *Li*, *i* ∈ {1, ..., *n*} được xác định là:

*Li* = {*C*1, *C*2, [*qi*1, ..., *qik*]} (7)

Trong đó *C*1 và *C*2 là hai thành phần được liên kết bởi Liên kết L. Các liên kết là một lớp đặc biệt, bởi vì các thành phần mà chúng kết nối nên là de ﬁ ned, tuy nhiên thuộc tính liên kết không phụ thuộc vào thuộc tính thành phần. Điều này được trình bày lại trong Hình 1 vì lớp Liên kết không bắt nguồn từ lớp Thành phần, nhưng được liên kết với nó.

* 1. *Quy trình làm việc và đánh giá chất lượng*

Do đó, một triển khai hoặc liên kết cụ thể được mô tả bởi một vectơ *Ai Di*. Trong ngữ cảnh tối ưu hóa, vectơ thuộc tính này được ánh xạ tới vectơ chất lượng *Qi* = *normi*(*Ai*) bằng cách áp dụng bản đồ chuẩn hóa tương ứng cho từng thuộc tính. Mỗi vector chất lượng *Qi* có một vectơ liên kết *wi* vớiđiều đó, đối với mỗi thuộc tính, hãy tính trọng số của nó. Một quy trình làm việc trừu tượng là biểu diễn của một chuỗi các dịch vụ được gọi (có thể song song) để đáp ứng một yêu cầu nhất định (đáp ứng một yêu cầu nhất định).



D

TAGDB

D

TAGDB

D D D

iELSSI iELSSI Extract

D D

TAGDB TAGDB

ExampIe TAG WorkfIow

∈

Abstract WorkfIow Wo

(BPMN Activity Diagram)

Hình 2: Ví dụ quy trình làm việc tóm tắt

. . .

. . .

Hình 2 cho thấy một ví dụ về quy trình làm việc tóm tắt được lấy từ trường hợp sử dụng TAG của bạn. Quy trình làm việc bao gồm việc gọi một phiên bản *iELSSI* (cho phép duyệt cơ sở dữ liệu TAG), một hoặc nhiều cơ sở dữ liệu TAG song song, một phiên bản dịch vụ *Trích xuất* (cho phép truy xuất dữ liệu) và một hoặc nhiều cơ sở dữ liệu TAG song song. Mỗi hình chữ nhật trong hình là một dịch vụ cần được thay thế bằng một triển khai khi tạo một tác phẩm cụ thể cho tác phẩm trừu tượng. Lựa chọn triển khai ngầm định bao gồm lựa chọn các liên kết, tức là các thuộc tính liên kết được tính đến khi xây dựng một quy trình làm việc cụ thể.

Về mặt hình thức, một *quy trình làm việc tóm tắt* *W*0 được định nghĩa là *W*0 = (*V*, *E*, *f*0) với (*V*, *E*) là một đồ thị có hướng và *f*0 : *V* → *C* là một chức năng ánh xạ mỗi triển khai đến dịch vụ của nó.

Một *quy trình làm việc cụ thể* là một ví dụ của một *quy trình làm việc tóm tắt*. Nó được coi là *W* = (*V*, *E*, *fd*) với (*V*, *E*) là một đồ thị có hướng và *fd* là một bản đồ lựa chọn triển khai sao cho *fd* : *V* → *D*. Một quy trình làm việc cụ thể *W* là hợp lý cho *W*0 nếu chức năng ánh xạ việc triển khai tới một dịch vụ tương thích với chức năng chọn một dịch vụ cho mỗi nút. Trong trường hợp không có các yêu cầu hoặc ràng buộc cụ thể, *W khả thi cho W*0 nếu nó là hợp lý cho *W*0, tức là tất cả các kết hợp có thể có về nguyên tắc đều được chấp nhận.

*Chất lượng* của mỗi lần triển khai *i* được tính như sau:

** (8)

trong đó *n* là số thuộc tính được áp dụng cho việc triển khai đó. Tương tự, *chất lượng* của mỗi liên kết *j* được tính như sau:

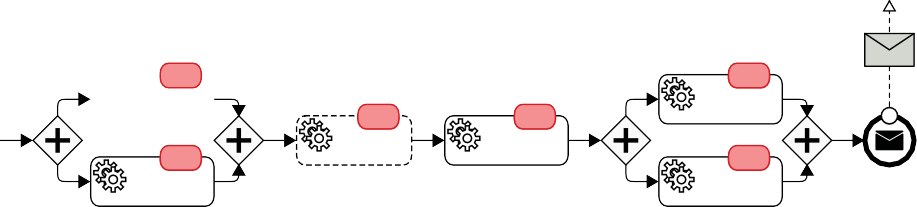
 (9)

trong đó *m* là số thuộc tính được áp dụng cho liên kết đó.

Như có thể thấy trong Hình 3, tổng quy trình làm việc không nhất thiết chỉ là một chuỗi. Do đó, chúng ta cần xác định vị trí các mẫu và các hàm tổng hợp thuộc tính tương ứng. Điều này đã được đề xuất trong công trình trước đây của chúng tôi [8]. Ở đây, chúng tôi tập trung vào các mẫu được sử dụng trong quy trình làm việc ví dụ của chúng tôi, như được mô tả trong Hình 3: trình tự và đồng bộ phân tách (song song).

Cho *p* = 1, ...*n* là các mẫu riêng biệt được sử dụng trong quy trình làm việc tóm tắt và *aggp* là hàm tổng hợp chất lượng cho mỗi mẫu. Tổng thể quy trình làm việc có chất lượng *Q*(*W*) được định nghĩa là:

 (10)



Sequence 1

D

iELSSI

ParaIIeI 1

D

TAGDB

Sequence 2

D

iELSSI

ParaIIeI 2

D

TAGDB

Seq. 3

D

Extract

D D

TAGDB TAGDB

ExampIe TAG WorkfIow

Hình 3: Ví dụ quy trình làm việc với các mẫu thành phần

Abstract WorkfIow Wo

(BPMN Activity Diagram)

. . .

. . .

# Tối ưu hóa Đa mục tiêu Động

Trong phần này, các hàm mục tiêu được mô tả và một thuật toán để giải quyết vấn đề tối ưu hóa được trình bày.

* 1. *Các chức năng mục tiêu*

Dựa trên mô hình chính thức được phát triển trong phần trước, bây giờ chúng ta có thể hình thành các chức năng khách quan và các ràng buộc của bài toán tối ưu hóa cụ thể của chúng ta. Như đã trình bày trong phần giới thiệu, vấn đề lựa chọn triển khai hiện tại được nghiên cứu từ hai góc độ khác nhau. Đầu tiên, từ quan điểm của người dùng, mục tiêu là giảm thiểu thời gian thực hiện của mỗi yêu cầu, đồng thời đáp ứng các ràng buộc như yêu cầu về tính khả dụng và độ tin cậy tối thiểu. Vì chúng ta quan tâm đến thời gian thực hiện của công việc khả thi liên quan đến nhau (tức là xếp hạng công việc khả thi) thay vì theo giá trị tuyệt đối, ánh xạ thời gian thực hiện với Xếp hạng chất lượng được giới thiệu trong Phần 4.3 là một sự trừu tượng hợp lệ. Thứ hai, từ quan điểm của nhà cung cấp hệ thống được cho là một phiên bản trung tâm, mục tiêu là đảm bảo việc sử dụng hợp lý và hiệu quả tất cả các tài nguyên có sẵn. Mục tiêu này được gọi là sử dụng phần còn lại. Do đó, trong quá trình tối ưu hóa, chúng tôi không chỉ xem xét một yêu cầu của người dùng mà còn xem xét hành vi tổng thể của hệ thống trong một khoảng thời gian nhất định. Về mục tiêu tối ưu hóa, điều này có thể được ánh xạ để giảm thiểu sự khác biệt của xếp hạng chất lượng cho mỗi tác phẩm trừu tượng. Do đó, chúng tôi đang phải đối mặt với một vấn đề tối ưu hóa đa mục tiêu bao gồm hai mục tiêu:

1. Tối đa hóa chất lượng của một yêu cầu duy nhất: *Max*(*Q*(*W*))

2. Giảm thiểu sự khác biệt của chất lượng theo thời gian: cho mỗi *W*0, *minVar*(*Q*(*W*)) có khả năng chịu các ràng buộc rõ ràng.

Hai mục tiêu này có ý nghĩa thống nhất, khi một mục tiêu cố gắng đạt được giải pháp tốt nhất cho một yêu cầu nhất định, mà không tính đến trạng thái hệ thống tổng thể và hậu quả đối với các yêu cầu khác có thể xảy ra trong cùng một khung thời gian và mục tiêu thứ hai có xu hướng tối ưu hóa yêu cầu phân phối và sự công bằng của việc sử dụng tài nguyên. Như trong hầu hết các bài toán tối ưu hóa đa mục tiêu, mục tiêu là tìm kiếm và điều tra một tập hợp các giải pháp thỏa mãn cả hai mục tiêu ở mức có thể chấp nhận được. Nói cách khác, mục tiêu là xác định các giải pháp trong *tập hợp tối ưu Pareto*.

Ngoài ra, chúng tôi yêu cầu khung tối ưu hóa tổng thể phải năng động trong các khía cạnh sau:

**Những thay đổi trong môi trường**. Môi trường chúng ta đang đối mặt rất năng động, vì các trang web, tài nguyên và triển khai có thể được thêm vào và các thuộc tính có thể thay đổi theo thời gian. Như đã đề cập trước đó, hệ thống không liên tục thay đổi và các thay đổi được kiểm soát tốt, nhưng việc tối ưu hóa cần phải thích ứng với chúng.

•

**Trọng lượng khách quan.** Trong khi thiết kế thuật toán tối ưu hóa, một quyết định cần thực hiện là về trọng số tương đối của mục tiêu này với mục tiêu kia. Trong môi trường ứng dụng nổi tiếng như của chúng tôi, chúng tôi tranh luận rằng chúng tôi có thể loại bỏ việc sử dụng phụ thuộc vào thời gian và đặc điểm ứng dụng chuyên nghiệp, ví dụ: thời gian sử dụng cao điểm. Tùy thuộc vào các nghiệp vụ này, chúng tôi muốn tập trung vào mục tiêu này hay mục tiêu khác. Ví dụ, trong các giai đoạn có hoạt động của hệ thống rất thấp, mục tiêu đầu tiên (tối đa hóa xếp hạng chất lượng) nên có trọng số cao hơn. Trong thời gian sử dụng hệ thống cao, điều quan trọng hơn là phải phân phối đều các yêu cầu, do đó mục tiêu thứ hai có trọng số cao hơn. Điều này chỉ có thể thực hiện được vì chúng tôi có thể xác định tỷ lệ này dựa trên việc khai thác dữ liệu lịch sử.

•

**Ưu tiên các công việc trừu tượng.** Trong một kịch bản ứng dụng thế giới thực, có thể là một yêu cầu ưu tiên một số trường hợp sử dụng hơn những trường hợp khác trong một khung thời gian xác định. Ví dụ: trong thời kỳ hoạt động phân tích dữ liệu cao (ví dụ: trước hội nghị) yêu cầu trích xuất dữ liệu, công việc tóm tắt liên quan đến dịch vụ Trích xuất có thể được ưu tiên hơn tất cả những công việc khác. Do đó, chúng tôi muốn cung cấp một cơ chế cho phép thiết lập các ưu tiên như vậy và sử dụng chúng trong quá trình tối ưu hóa.

•

Dựa trên các yêu cầu động này, có thể xác định một số chuyên gia mô tả trạng thái hệ thống tại một thời điểm nhất định.

* 1. *Thuật toán di truyền dựa trên bộ nhớ rõ ràng*

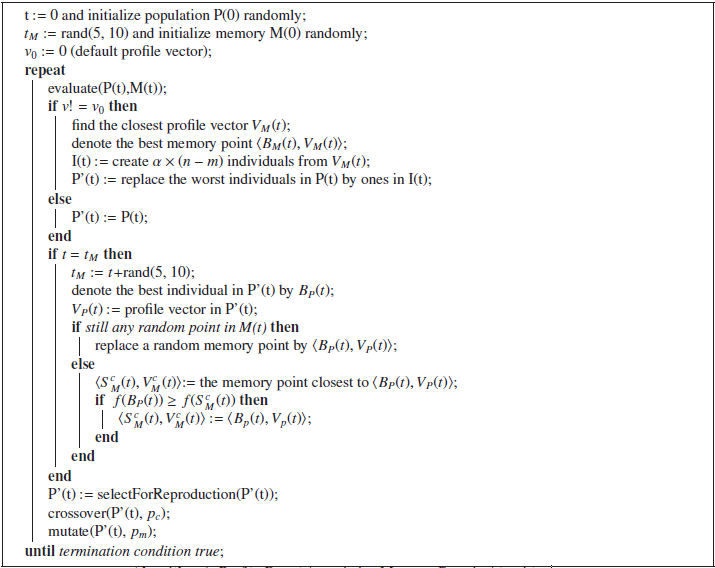
Thuật toán di truyền (GA) rất phù hợp để giải các bài toán tối ưu hóa đa mục tiêu (DMO) tĩnh cũng như động, như đã nêu trong [9, 10]. Trong các vấn đề DMO, điều mong muốn là có thể phản ứng nhanh với các thay đổi mà không cần bắt đầu quá trình tối ưu hóa từ đầu. Về mặt GA, điều này có nghĩa là tập hợp của thuật toán phải phản ứng với các thay đổi càng nhanh càng tốt [9]. Chủ yếu tồn tại bốn cách tiếp cận để đạt được mục tiêu này, theo [11]: đa dạng sau các thay đổi, đa dạng trong thời gian chạy, kỹ thuật dựa trên bộ nhớ và kỹ thuật đa dân số.

Trong trường hợp sử dụng hiện tại, các kỹ thuật dựa trên bộ nhớ là những kỹ thuật hứa hẹn nhất, vì chúng ta đang phải đối mặt với một vấn đề trong đó các điều kiện tương tự xuất hiện trở lại trong khoảng thời gian đều đặn (thời gian / mức sử dụng). Có hai cách lưu trữ thông tin trong bộ nhớ, bằng bộ nhớ ngầm hoặc bộ nhớ tường minh [11]. Các lược đồ bộ nhớ ngầm sử dụng các biểu diễn dư thừa để lưu trữ ngầm định thông tin có thể được sử dụng trong quá trình chạy. Các lược đồ bộ nhớ rõ ràng cho phép lưu trữ thông tin hữu ích từ thế hệ hiện tại để sử dụng lại trong các thế hệ hoặc môi trường sau này. Do đó, chúng sử dụng thêm không gian lưu trữ. Có thể không chỉ lưu trữ các giải pháp tốt từ thế hệ hiện tại, mà còn cả thông tin liên kết các giải pháp đó với một môi trường cụ thể, trong trường hợp của chúng tôi là ưu đãi sử dụng. Trong các lần chạy sau, các đặc điểm môi trường có thể được khớp với thông tin môi trường đã biết và các giải pháp tốt liên quan có thể được truy xuất và cập nhật. Đây được gọi là lược đồ bộ nhớ liên kết [12]. Vì chúng ta đang phải đối mặt với một môi trường được kiểm soát có thể được mô tả tại bất kỳ thời điểm nhất định nào, nên một chiến lược tối ưu hóa sử dụng thuật toán di truyền dựa trên bộ nhớ rõ ràng là đầy hứa hẹn.

Một ví dụ mã giả đơn giản của một thuật toán như vậy được liệt kê trong Thuật toán 1. Cơ sở tri thức được liên kết với môi trường tối ưu hóa. Trong các khoảng thời gian đều đặn, cá thể tốt nhất của một tập hợp được lưu trữ cùng với một vectơ đặc trưng cho trạng thái hệ thống. Mỗi khi vectơ này thay đổi, tức là gặp một môi trường mới, thuật toán di truyền được khởi động lại với một phần nhỏ các cá thể ngẫu nhiên nhất định và một phần nhỏ các cá thể thành công nhất định cho một môi trường tương tự. Chìa khóa của cách tiếp cận này là biểu diễn và lưu trữ một cách hợp lý thông tin môi trường và ánh xạ tương đồng giữa các môi trường khác nhau. Môi trường có thể được mô tả bằng cách sử dụng vectơ hồ sơ được mã hóa bit v có độ dài và mô tả ﬁ xed. Giá trị 1 tại một vị trí nhất định có nghĩa là chương trình ưu đãi được liên kết hiện đang hoạt động, nếu không giá trị được đặt thành 0. Thời gian và mức độ sử dụng có thể được tính toán bằng cách phân tích dữ liệu ghi nhật ký lịch sử. Định nghĩa của chuyên gia đối với hệ thống kiểm định về mặt triển khai và trạng thái của chúng cần được điều tra thêm.

Đặc biệt chúng ta phải chú ý đến hai chức năng GA đặc biệt đã đề cập: giao nhau và đột biến và cách chúng có thể được áp dụng cho việc lựa chọn các thành phần công việc của chúng ta.

* Hàm **kết hợp** lấy hai hoặc nhiều cá nhân - công việc cụ thể với các triển khai đã chọn - của một quần thể nhất định và kết hợp chúng để suy ra một cá thể mới và (hy vọng) tốt hơn. Như đã đề cập trong phần trước, chúng tôi đo lường chất lượng của một cá nhân - một giải pháp có tác dụng đáp ứng mục tiêu tối ưu hóa của chúng tôi - bằng Q (W). Về thách thức lựa chọn dịch vụ của chúng tôi, chúng tôi có thể áp dụng chức năng kết hợp bằng cách chia nhỏ công việc thành một số công việc con và kết hợp chúng lại tại một liên kết nhất định, tức là chúng tôi chia công việc ví dụ sau Khối song song 1 và trước khối Trình tự 2. Do đó, mỗi tác phẩm có thể được chia thành hai nửa, cho phép xây dựng một tác phẩm mới (cá nhân) từ phía trước và phía sau của hai tác phẩm phân biệt (không thể phân biệt).
* Hàm **đột biến** cho phép thay đổi - thay thế tương ứng - các đặc điểm của một cá thể bằng cách sử dụng một đặc tính có thể so sánh, tương tự để đạt được giá trị chất lượng cao hơn so với đặc điểm được thay thế. Do sự cải thiện này, chất lượng tổng thể của cá nhân tăng lên. Được áp dụng cho một tác phẩm trong kịch bản lựa chọn dịch vụ của chúng tôi, điều này có nghĩa là chúng tôi chỉ thay thế một hoặc nhiều đoạn của một tác phẩm đã xây dựng (riêng lẻ). Do đó chúng ta phải xác định các triển khai có các đặc điểm tương tự liên quan đến logic nghiệp vụ, nhưng có mức chất lượng cao hơn. Như trong trường hợp chéo, mục tiêu của chức năng này là tạo ra những cá nhân mới (gu làm việc) thể hiện chất lượng cao hơn những cá nhân hiện có thực tế.



**Thuật toán 1:** Thuật toán di truyền bộ nhớ liên kết dựa trên hồ sơ

# Các công việc liên quan

Các thuật toán di truyền gần đây đã được tận dụng để giải quyết các vấn đề về thành phần dịch vụ nhận biết QoS [13, 14, 15, 16]. Trong khi một số giải quyết các vấn đề tối ưu hóa đơn mục tiêu [14], các cách tiếp cận khác giải quyết các vấn đề đa mục tiêu. Wada et. al. [15] áp dụng thuật toán di truyền đa mục tiêu cho xác suất thành phần dịch vụ nhận biết QoS. Họ cũng xác định các dịch vụ (Dịch vụ trừu tượng), triển khai (Dịch vụ cụ thể), công việc trừu tượng (Quy trình nghiệp vụ), công việc cụ thể (Phiên bản quy trình nghiệp vụ) và xem xét ba thuộc tính QoS ví dụ, đó là thông lượng, độ trễ và chi phí. Họ thiết lập một quy trình kinh doanh dựa trên ba mẫu với các chức năng tổng hợp liên quan cho các thuộc tính QoS đã đề cập. Dựa trên các khái niệm này, các tác giả trình bày một thuật toán di truyền đa mục tiêu được gọi là E3-MOGA, trong đó mỗi cá nhân đại diện cho một giải pháp của bài toán thành phần dịch vụ. Giá trị ﬁ tness của nó sau đó được tính toán dựa trên thứ hạng và mật độ thống trị của nó. Các cá nhân tiến hóa thông qua sự giao nhau và đột biến gen, và cuối cùng, thuật toán đưa ra một tập hợp các giải pháp Pareto. Trong mỗi lần lặp lại, ba mức dịch vụ khác nhau được xem xét. Cách tiếp cận được áp dụng gần với phương pháp của chúng tôi, nhưng nó không tính đến động lực của hệ thống cơ bản và giả định các chức năng mục tiêu tĩnh. Qiqing et. al. [16] trình bày một cách tiếp cận để tối ưu hóa chất lượng dịch vụ trên toàn cầu bằng cách sử dụng tối ưu hóa đàn kiến ​​đa mục tiêu (MOCACO). Các tác giả so sánh thuật toán của họ với thuật toán di truyền đa mục tiêu (MOGA) được đề xuất trong [13] cho cùng một loại vấn đề và cho thấy rằng MOCACO vượt trội hơn MOGA cả về hiệu suất thời gian chạy và về việc đưa ra nhiều giải pháp tốt hơn. Một lần nữa, hệ thống cơ bản được giả định là tĩnh, điều này khác với cách tiếp cận của chúng tôi.

# Kết luận và những công việc trong tương lai

Bắt đầu từ một bài toán tối ưu hóa trong thế giới thực như được mô tả trong Phần 2, chúng tôi đã phát triển một bản đại diện chung của hệ thống phân tán đã nghiên cứu, suy ra một mô hình chính thức và xây dựng một bài toán tối ưu hóa đa mục tiêu động cụ thể. Phân tích vấn đề tối ưu hóa chi tiết hơn dẫn đến kết luận rằng một thuật toán di truyền dựa trên bộ nhớ rõ ràng là một cách tiếp cận giải pháp thích hợp. Các bước tiếp theo bao gồm việc triển khai thuật toán được mô tả, hiệu chỉnh mô hình và đánh giá các biến thể có thể có của thuật toán. Sự chú ý đặc biệt sẽ được dành cho sự biểu diễn e cient của môi trường trong bộ nhớ. Việc đánh giá cách tiếp cận tổng thể và đánh giá mô hình của chúng tôi sẽ được thực hiện cả trong hệ thống thực và trong môi trường mô phỏng. Một sổ đăng ký dịch vụ được đặt ra để lưu trữ tất cả các thành phần của hệ thống TAG phân tán và tính toán các thuộc tính cụ thể như được mô tả trong Phần 3. Việc triển khai nguyên mẫu của thuật toán lựa chọn triển khai sẽ được cung cấp và đánh giá trong hệ thống sản xuất. Song song, để nghiên cứu khả năng mở rộng của cách tiếp cận, chúng tôi dự định xây dựng một bản sao mô phỏng của môi trường của chúng tôi trong công cụ SimGrid [17], mô phỏng các kịch bản tải khác nhau và đánh giá lại thuật toán dựa trên các kết quả này. Ngoài ra, như được mô tả trong [9], xử lý song song nói chung có thể tăng tốc thời gian thực hiện các thuật toán di truyền được áp dụng cho các bài toán tối ưu hóa đa mục tiêu động. Do đó, chúng tôi có kế hoạch cung cấp một phiên bản song song của thuật toán được đề xuất.

# Tài liệu tham khảo

1. The ATLAS Collaboration, The ATLAS Experiment at the CERN Large Hadron Collider, in: JINST 3 S08003, 2008, p. S08003.
2. The ATLAS Collaboration, ATLAS computing, Technical Design Report ATLAS, CERN, Geneva, 2005.
3. T. Yu, Y. Zhang, K.-J. Lin, Eﬃcient algorithms for Web services selection with end-to-end QoS constraints, ACM Trans. Web 1 (1) (2007) 6.
4. D. Pisinger, Algorithms for Knapsack Problems, Ph.D. thesis, DIKU, University of Copenhagen, Denmark, Technical Report 95-1 (1995).
5. E. Vinek, P. P. Beran, E. Schikuta, Mapping Distributed Heterogeneous Systems to a Common Language by Applying Ontologies, in: PDCN 2011: Proceedings of the 10th IASTED International Conference on Parallel and Distributed Computing and Networks, IASTED/ACTA Press, 2011, accepted for publication.
6. Distributed Management Task Force, Common Information Model (CIM), Schema 2.28.0 (February 2011).

URL <http://dmtf.org/standards/cim>

1. K. Koﬂer, I. Ul Haq, E. Schikuta, A Parallel Branch and Bound Algorithm for Workﬂow QoS Optimization, Parallel Processing, International Conference on 0 (2009) 478–485.
2. E. Vinek, P. P. Beran, E. Schikuta, Classiﬁcation and composition of qos attributes in distributed, heterogeneous systems, in: submitted to CCGRID, 2011.
3. M. Ca´mara, J. Ortega, F. de Toro, A single front genetic algorithm for parallel multi-objective optimization in dynamic environments, Neuro- comput. 72 (2009) 3570–3579.
4. A. Konak, D. W. Coit, A. E. Smith, Multi-objective optimization using genetic algorithms: A tutorial, Reliability Engineering & System Safety 91 (9) (2006) 992 – 1007, special Issue - Genetic Algorithms and Reliability.
5. Y. Jin, J. Branke, Evolutionary Optimization in uncertain Environments - A Survey, Evolutionary Computation, IEEE Transactions on 9 (3) (2005) 303–317.
6. S. Yang, Explicit Memory Schemes for Evolutionary Algorithms in Dynamic Environments, in: S. Yang, Y.-S. Ong, Y. Jin (Eds.), Evolution- ary Computation in Dynamic and Uncertain Environments, Vol. 51 of Studies in Computational Intelligence, Springer Berlin / Heidelberg, 2007, pp. 3–28.
7. S. Liu, Y. Liu, N. Jing, G. Tang, Y. Tang, A Dynamic Web Service Selection Strategy with QoS Global Optimization Based on Multi-objective Genetic Algorithm, in: H. Zhuge, G. Fox (Eds.), Grid and Cooperative Computing - GCC 2005, Vol. 3795 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2005, pp. 84–89.
8. M. C. Jaeger, G. Mhl, QoS-based Selection of Services: The Implementation of a Genetic Algorithm, in: T. Braun, G. Carle, B. Stiller (Eds.), In KiVS 2007 Workshop: Service-Oriented Architectures und Service-Oriented Computing (SOA/SOC, VDE Verlag, Bern, Switzerland, 2007, pp. 359–370.
9. H. Wada, P. Champrasert, J. Suzuki, K. Oba, Multiobjective Optimization of SLA-Aware Service Composition, in: Proceedings of the 2008 IEEE Congress on Services - Part I, SERVICES ’08, IEEE Computer Society, Washington, DC, USA, 2008, pp. 368–375.
10. F. Qiqing, P. Xiaoming, L. Qinghua, H. Yahui, A Global QoS Optimizing Web Services Selection Algorithm Based on MOACO for Dynamic Web Service Composition, Information Technology and Applications, International Forum on 1 (2009) 37–42.
11. H. Casanova, A. Legrand, M. Quinson, SimGrid: A Generic Framework for Large-Scale Distributed Experiments, in: Proceedings of the Tenth International Conference on Computer Modeling and Simulation, IEEE Computer Society, Washington, DC, USA, 2008, pp. 126–131.