Studies in Computational Intelligence, 66, pg. 229-248

Evolutionary Computing within Grid Environment

Ashutosh Tiwari, Gokop Goteng and Rajkumar Roy

Studies in Computational Intelligence, 66, pg. 229-248

Decision Engineering Centre, Manufacturing Department, School of Applied Sciences,
Cranfield University, Cranfield, Bedfordshire, MK43 0AL, UK
{*a.tiwari, g.l.goteng, r.roy*}*@cranfield.ac.uk*
http://www.cranfield.ac.uk/sas/mem/decisionengineering/

Abstract. Evolutionary computing (EC) techniques such as genetic algorithm (GA), genetic programming (GP), evolutionary programming (EP) and evolution strategies (ES) mimic nature through natural selection to perform complex optimisation processes that require more than one solutions. Grid-enabled environment provides suitable framework for EC techniques due to its computational and data capabilities. In addition, the semantic and knowledge Grids aid in the design search and exploration for multi-objective optimisation tasks. This chapter explores some problem solving environments such as Geodise (Grid-Enabled Optimisation Design Search for Engineering), FIPER (Federated Intelligent Product Environment), SOCER (Service-Oriented Concurrent Environment), DAME (Distributed Aircraft Maintenance Environment) and Globus toolkit to demonstrate how EC techniques can be performed more efficiently within Grid environment. Service-oriented and autonomic computing features of Grid are discussed to highlight how EC algorithms can be published as services by service providers and used by service requestors dynamically. Grid computational steering and visualisation are features that can be used for real-time tuning of parameters and visual display of optimal solutions. This chapter demonstrates that grid-enabled evolutionary computing marks the future of optimisation techniques.

1 Introduction

The use of Evolutionary computing (EC) to solve many optimisation problems is still hindered by the inadequate computational power and data manipulation platform for computational and data intensive problems. Examples of such computational and data intensive applications are engineering design optimisation, bioinformatics, pharmaceutical and particle physics simulations and many others. Additionally, since EC techniques mimic nature through natural selection, knowledge-driven problem solving environments (PSEs) are ideal frameworks for optimisation problems using EC. Grid computing (GC) promises to provide computational resources and services for optimisation processes using EC techniques. Besides, the service-oriented architecture (SOA) in Grid-based deployments provide easy and seamless access to heterogeneous distributed optimisation codes to users in a dynamic and coordinated manner. Service-oriented problem solving environments (SO-PSEs) such as Geodise (Grid-Enabled Optimisation Design Search for Engineering), FIPER (Federated Intelligent Product Environment) and SORCER (Service-Oriented Concurrent

Environment) are good examples of Grid-based PSEs that use semantic Grid and ontology to intelligently drive optimisation processes. Globus toolkit which serves both as a middleware as well as a PSE has been the major innovative contribution to the second generation Grid by the Global Grid Forum (GGF) which has now been merged with the Enterprise Grid Alliance (AGA) to become the Open Grid Forum (OGF).

This chapter will focus on how evolutionary computing techniques can exploit Grid-based framework to provide seamless access to optimisation resources and services for users involved in multidisciplinary optimisation applications. The first part of this chapter will discus evolutionary computing and related application areas. The second part will look at intelligent Grid environments such as semantic Grid, knowledge Grid and autonomic computing. The third part will concentrate on service-oriented and problem solving environments for Grid-based problems. The fourth part will delve on problem solving environment case studies. The last part of this chapter will talk about collaborative evolutionary computing.

2 Evolutionary Computing

Evolutionary computing uses techniques such as genetic algorithms (GAs), genetic programming (GP), evolutionary programming (EP) and many others. These techniques work on the principles of natural selection based on Darwin's theory of evolution. This principle works on the composition of genetic traits called chromosomes, in which successive operations through crossover or mutation give rise to better performing off-springs (population) due to successive refinement of these hereditary traits. In the same way, evolutionary computing techniques mimic this phenomenon of natural selection to improve upon classical methods of optimisation such as preference-based and generating methods that work well only on single-objective optimisation problems [10]. With evolutionary computing techniques, multi-objective optimisation problems can be solved, producing multiple optimal solutions in a single simulation run. This is not possible with classical optimisation methods. However, the evolutionary computation can be computationally expensive for complex optimisation problems. Imagine an optimisation problem of an underground system with hundred of thousands of optimal solutions. This is where Grid-based framework comes in. Grid computing is a distributed computing infrastructure which aims to overcome large-scale computation within the fields of science, engineering and business with its secured, pervasive, persistent, dynamic and coordinated computational resources and services [13].

Evolutionary programming which was introduced by Fogel is an attempt to create artificial intelligence (AI) using finite state machines (FSM) to predict future events on the basis of previous behaviours of elements during successive transitions [3]. FSM is an abstract machine which converts input symbols into sequence of output results depending on some finite set of states and finite state of transition rules. With this in mind, the Semantic Grid, which is an extension of the Grid that gives well-defined meaning to processes to ensure reuse in subsequent similar applications, is a good tool to capture optimisation processes for future use within evolutionary

computing domains. Other evolutionary algorithms such as GA and evolutionary strategies work in similar ways that may need semantic Grid as knowledge based framework for intelligent application of evolutionary computing used in various real world problems and experimental simulations of random and optimised stimulus sequences over a range of design parameters [38]. This chapter will take a look at intelligent Grid environments and later discuses service-oriented and problem solving environments as they relate to evolutionary computing applications. Figure 1 is a simple diagram of GA process.

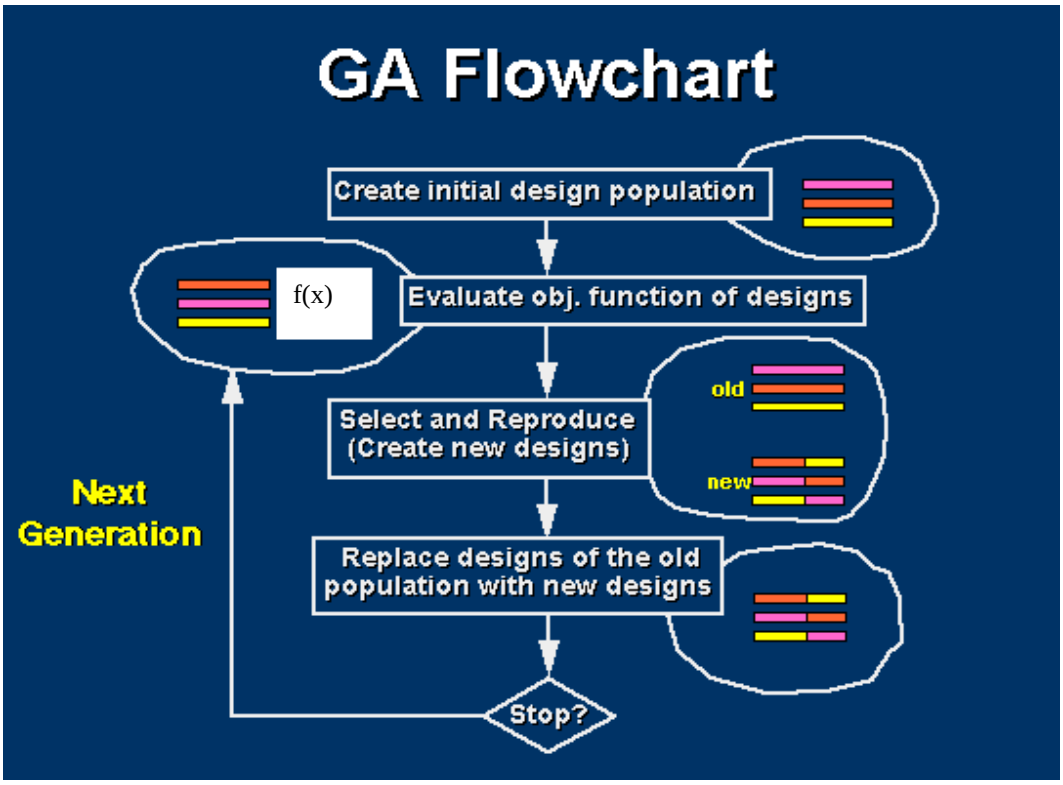


Figure 1: Genetic Algorithm (GA) Flowchart

3 Intelligent Grid Environment

Evolutionary computing algorithms produce explosion of solutions in each iteration which require intelligent analysis to select the optimum solution. Again, it is

important to capture optimisation processes and reuse them for new optimisation tasks or to serve as a guide to new optimisation engineers. The use of ontology in semantic and knowledge Grids is important in creating an intelligent problem solving environment for evolutionary computing applications. Ontology is the structured representation of the concepts, functions and relationships of the building blocks of a special field of knowledge using high-level syntax common to the users of that field of knowledge.

3.1 Semantic and Knowledge Grids

Research activities usually involve multidisciplinary experts working together for collaboration. Dynamic coordination and resource sharing is very important in this arrangement. Also of great importance is how each expert will understand what others are doing and how this affects the overall aim of the group. However, dynamic coordination and resource scheduling is a big challenge coupled with knowledge representation issues that can be understood by all cooperating experts. Semantic Grid aims at capturing the processes of activities of domain experts and publishing it for reuse by others in an intuitive manner. Knowledge Grid on provides management services for processes produced from semantic Grid using Ontologies through rich information service protocols [32]. Knowledge Grid enables knowledge reuse through description, discovery and access to resources using protocols such as Grid Services Description Language (GSDL) and Universal Description, Discovery and Integration (UDDI) services.

Evolutionary computing creates population of solutions with repeated iterations. Thus, more data is produced which requires intelligent method of analysis to select most suitable optimum solution using multiple trade-offs. This process of obtaining the optimum solution in the presence of multiple trade-offs is better achieved through semantic Grid which uses ontology to explicitly specify conceptualisation where definitions associate concepts, functions, context, taxonomies and relationships with high-level (human-readable) text as well as low-level (machine-readable) axioms [20]. The high-level text serves as an intuitive inference point and knowledge reuse for collaborators can be achieved while the low-level is important for communication among computational resources as well as among the collaborators. The layered architecture of the Grid provides a convenient way of representing human-readable optimisation rules and processes and machine-readable axioms on different layers. For example, to present a transparent view to users, semantic and knowledge-based rules should fall within the application level [33]. [33] demonstrated this way of using intelligent systems in their N2Grid project which uses neural networks to exchange information and exploit available computational resources using a 3-layered architecture. The layers are knowledge layer which uses 2-dimensional information for analysis and mining of data to provide problem solving mechanisms, information layer which uses 1-dimensional information for semantic interpretation and data layer which provides data administration.

Intelligent knowledge-based layered architecture for the Grid is the next-generation of future Grid implementation using service-oriented models [6]. Service-

oriented architectures will be discussed later. Additionally, the semantic Grid seeks to incorporate its predecessor, the semantic Web, which is an extension of the Web in which information is given well-defined meaning [5]. Just like the semantic Web, the semantic Grid needs an expressive and extensible way of describing Grid functionalities at all levels using flexible mechanisms to explore trade-offs in Grid's complex decision space that incorporates heuristics and constraints into optimisation and computational processes of evolutionary computing techniques [15].

3.2 Autonomic Computing

Grid computing is a global infrastructure for coordinated and dynamic platform for large-scale science and business activities [13]. This global infrastructure will grow at a rate not imaginable with many heterogeneous interacting hardware, software, services and people. In this case, managing the infrastructure may become almost impossible with conventional system management and scheduling algorithms. Autonomic computing is a concept to address this complexity in managing hundreds of thousands of heterogeneous Grid components with each accessing and using other services and resources as well as others using it as their resource or service in a symbiotic set up [27]. The ultimate goal is to build a Grid resource management system within every Grid service that can automatically manage itself and adapt to its changing environment even in the presence of errors or other harsh conditions through autonomic configurations and reconfigurations. An agent-based or master-worker paradigm is an innovative way to view the autonomic framework [16]. Arguably, security issues concerning the protection of each service provider in the Grid may hinder this autonomic goal of resource management from a global point of view. The most innovative way to overcome this is through the single-sign on proxy protocol found in middleware service such as Globus. This will allow resources for evolutionary computation to be used from multiple administrative domains among multiple workloads as the middleware is called upon to authenticate and authorise the integration of services through some adaptive autonomic concepts [22]. [36] demonstrated that autonomic computing can allow resource providers to collaborate while maintaining autonomy over their individual codes. [36] used genetic algorithm optimisation logic as Grid service at Southampton University in UK to drive the design search process while the analysis code was located in Singapore to evaluate the objective functions. The synergistic effect of computational powers at both locations also helped in speeding up the identification of an optimum minima using evolutionary algorithm such as the GA. Dynamic composition of these resources is also very important to synthesise available heterogeneous resources and services based on conflicting objectives and constraints during optimisation processes [2].

4 Grid-Based Architectures

The layered architecture of the Grid provides a convenient way for service providers to publish services based on high-level (human-dependent) or low-level (machine

dependent) requirements regardless of the middleware used. An extension to the conventional architecture is the service-oriented architecture (SOA). SOA together with Grid-enabled problem solving environments (PSEs) provide a collaborative framework for virtual organisations to work together and have seamless access to resources and services dynamically in a coordinated manner. SOA will allow programmers to provide different EC codes as services that can be used by design engineers for optimisation processes regardless of location. An evolutionary algorithm may work well for a particular optimisation problem and performs poorly for another problem [10]. In this case, SOA provides a rich pool of different evolutionary algorithms as services. Autonomic computing helps to automate the selection process of which algorithm is best suited for a particular optimisation problem based on the objective functions and constraints. PSEs offer platform for easy access to EC tools and services for computation, data, collaboration and resource management. This is important because EC techniques are meant to solve a variety of optimisation problems and this gives many users who need different optimisation algorithms can seamlessly have access to the services.

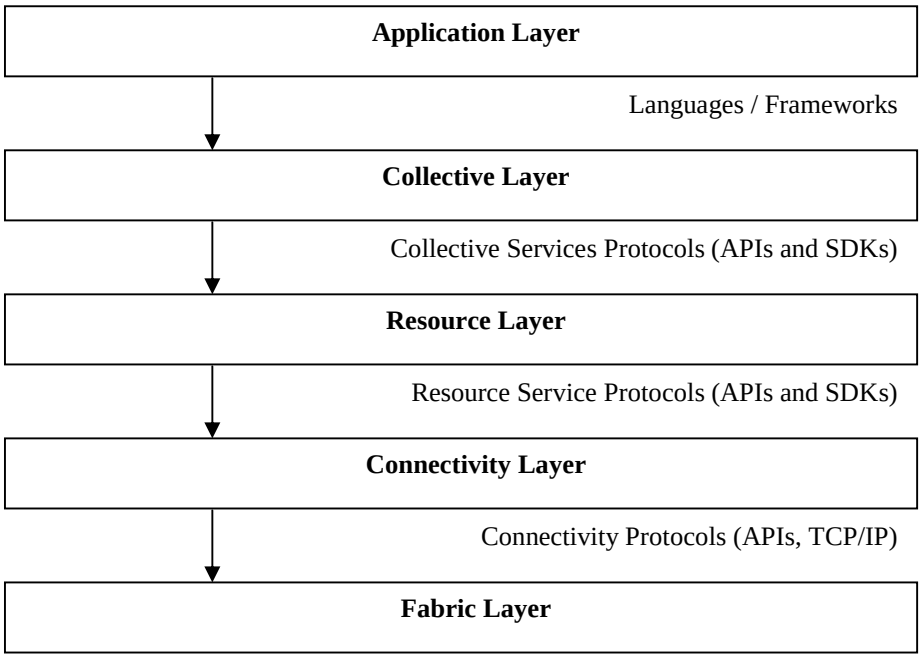


Figure 2: Grid Computing Architecture [1].

Figure 2 is the generic architecture of grid computing. The layered architecture allows different resources to be organised and published at different layers. For example, the lowest layer (Fabric Layer) is meant to have physical resources such as computers, sensors, instruments and core operating systems while the topmost layer (Application Layer) has resources that interact directly with end users. Such resources include application software like CAD (computer aided design) systems. The

connectivity layer ensures network connectivity through TCP/IP (Transport Control Protocol/ Internet Protocol) and resource layer publishes APIs (Application Programmers Interfaces) and SDKs (Software Development Kits) for grid programmers. GA programmers can use the application layer and resource layer.

4.1 Service-Oriented Architecture

To understand service-oriented architectures (SOAs), it is important to trace the history of the electricity power Grid from which the computational Grid derived its name. Before 1910, there was electricity, but the supply and usage was in crude forms. To use electricity power whether for domestic or industrial purpose, one needed to construct, own and maintain a power generating plant. With this form of electricity generation, there was not enough power to drive heavy machineries and this hampered the invention and production of many devices that needed electricity power to operate and function. Intermittent power failures and the cost of constructing and maintaining the generating plant added to this problem. This trend changed after 1910 when the electricity power Grid was invented. With the electricity power Grid, users do not need to construct, own and maintain a generating plant. All users need to do is to subscribe for the utility (electricity power service) and 'plug and use' the power transmitted from any source (hydro, solar, etc). In this way, electricity became cheaper and more reliable. This gave incentive to scientists and technologists to invent and fabricate many devices that can use electricity. The electricity power sector became a big multi-billion dollar industry and contributed greatly to the growth of the economy. This is because it is now a service-oriented electricity power where generation, maintenance, transmission and distribution are essential components of the utility. This means that electricity power was itself not a problem, but the problem was lack of 'Grid' with its associated distribution and transmission features [13].

In the same way, computational scientists having experienced similar bottlenecks with computational power in applications such as CFD (computational fluid dynamics) and engineering design optimisation, are beginning to look towards the 'Grid' to do the same for them just as it did for the electrical power engineers and scientists in 1910. By construing the computational power Grid, computational resources can be published by service providers and consumed by service requestors (users) who are then charged per usage. This means that with Grid, application users may not need to own software systems in the future but subscribe to them and 'plug and use'. Figure 3 shows a simple arrangement of how dynamic Grid coordination takes place between service providers and service requestors.

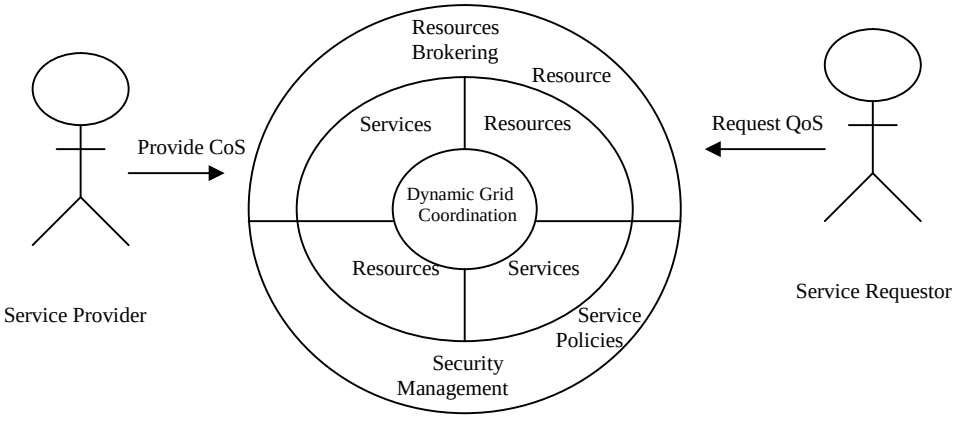


Figure 3 Resource Brokering between Service Providers and Service Requestors. Note: Cost of Service =CoS, Quality of Service =QoS

The above analogy between electricity utility and computational utility may sound good. However, service-oriented computational Grid cannot be as easily achievable as service-oriented electricity Grid. This is because the vision of the computational Grid is ambitious due to heterogeneous and dynamic coordination of hardware, software, sensors, expensive instruments, visualisation systems and people. The charge rate per service may not be as static as in electricity; it will include properties such as quality of service (QoS), time of services (ToS) and cost of service (CoS) may be dynamic in nature. The middleware issues and security issues involved in sharing knowledge related resources and information are complicated. These issues were not part of electricity power Grid. This means that computational scientists need to do much more than what was done to realise the electricity power Grid. Ownership and deployment of legacy codes such as GAs within SOA involve third generation Grids using OGSA (Open Grid Services Architecture) and WSRF (Web Services Resource Framework) platforms [18]. Grid computing aims to satisfy not only static physically distributed users but even mobile users can access resources dynamically. The DAME (Distributed Aircraft Maintenance Environment) project uses this concept to troubleshoot and maintain aircrafts having engine problems while flying through knowledge-based Grid services [23]. This makes SOA a dynamic interacting for reorganising software applications to adapt to static and mobile application coordination [24]. This location neutrality and technology neutrality between tightly coupled and loosely coupled systems are presenting a big challenge to SOA implementation. Evolutionary computing stands to benefit from this architecture because many different algorithms can be published as services and users can subscribe for the optimisation algorithm that best fits their needs. In addition, one does not need to own the optimisation codes, just subscribes for it and pay per usage. Total cost of ownership (TCO) of evolutionary computing tools which involves development, maintenance and upgrades is drastically reduced. There will be services for evolutionary computations, data management of evolutionary results, middleware for handling interoperability among different objective functions for multi-objective optimisation processes, visualisation services for output, collaboration services where optimisation expertise are distributed across geographical locations and modeling and search services. [26] developed a service-oriented system for design optimisation using genetic algorithm (GA), simulated annealing (SA) algorithm and Tabu Search (TS) algorithm. The system has data and parametric model located in different sites of

the Grid connected by a shared knowledge repository which allows a particular algorithm to be selected at a time.

4.2 Problem Solving Environments

To better understand and appreciate the role of Grid-enabled problem solving environments (PSEs) within multidisciplinary context, two scenarios will be described here. The first scenario is in the automobile industry. Consider that an automobile company is to launch a new car into the market soon and has concluded the design phase and is in the process of contacting its suppliers to bid for the components. It is important to note here that no single car manufacturer produces all the components required to manufacture a complete car from engine, wheels and light-fittings to windscreen. Most components are sub-contracted to suppliers. So, each competing supplier is likely to make changes or alterations to the original design sent to it because is not feasible or too costly or offer a better design at a cheaper rate or give other reasons to the original equipment manufacturer (OEM). The OEM will now look at the observations made by the suppliers and take a decision. This process of refining the design will be repeated until both the OEM and its suppliers finally agree. This process is not as straight-forward as described. It takes a lot of documentation, meetings, approvals, decision-making and many others. It is time consuming. Remember the earlier a new product gets to the market the better for the manufacturer because there other competitors who are equally working hard to get a better share of the same product in the market. Imagine that the OEM has 100 suppliers and these suppliers are scattered in different geographical locations around the world. This means spending a lot of time attending to suppliers. It is in this light that PSEs can help bring together both the OEM and its suppliers together within a Grid-enabled virtual organisation (VO) to work together in cooperation. This type of Grid platform is called Extraprise Grid [1].

One may ask the question why Grid since there are PDM (product data management) systems and collaboration packages around. The need for Grid-enabled PSE in this case is because it requires computational steering (making changes on one part of the design dynamically) and observing the effects on the whole designs using Grid visualisation systems. Every supplier can see the effect of changing a parameter or requirement and each supplier will better appreciate the goal of the manufacturer. When all stakeholders in the business have a common goal, the chance of rolling-out a good product is enhanced. Besides, the PSE provides easy access to both the OEM and suppliers with design optimisation codes and other computational services through Grid portals to allow seamless access regardless of geographical location. The computation of the different objective functions are carried out at different supplier's sites, making it faster. Different evolutionary algorithms are also used for the different components and the results shared. Figure 4 shows a hypothetical car manufacturer located in the U.S. with several of its car parts supplied by its partners distributed around the world. In this arrangement, Grid platform is a good environment for coordinated collaboration.

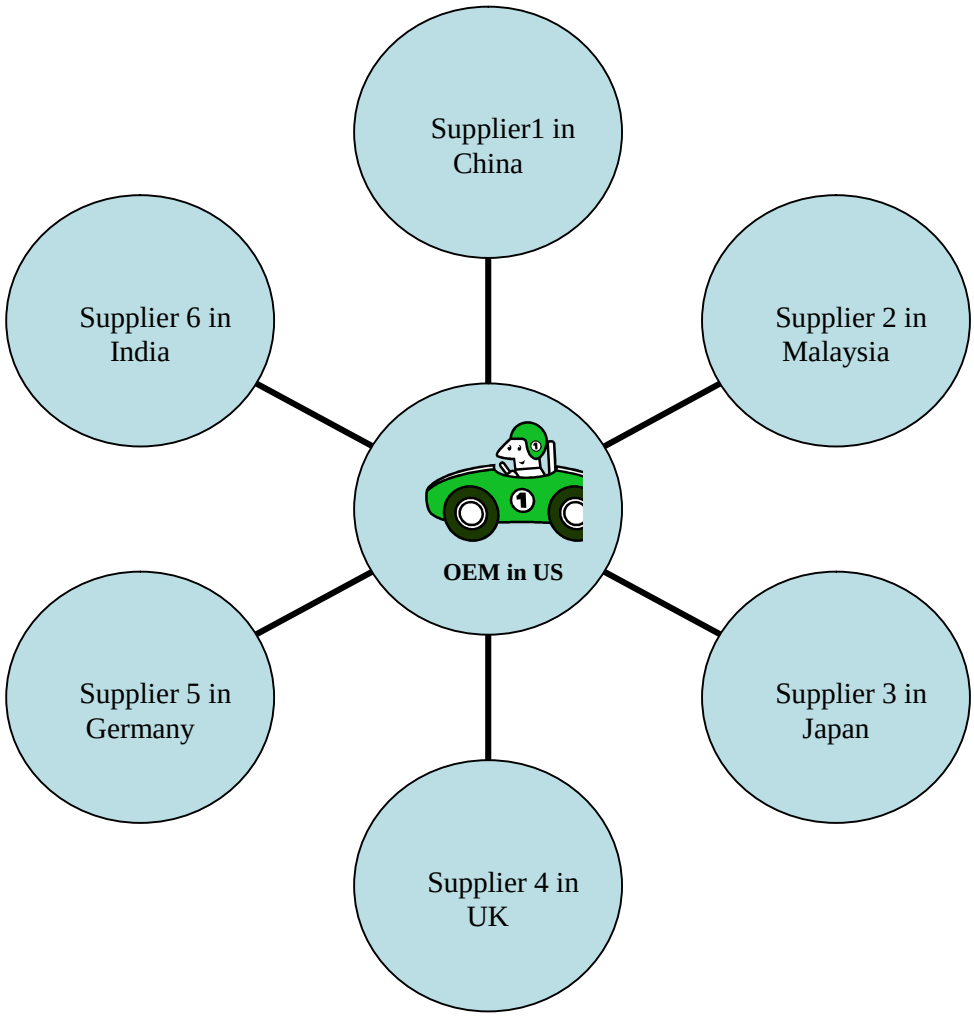


Figure 4 A Hypothetical Car Manufacturer (OEM) with many Suppliers of its Car Parts Located around the World

Using the OEM/supplier scenario, evolutionary computation for different objective functions can be carried out at different geographical locations (suppliers) and the results converged at the centre of decision making (OEM). In this way, the computation will be faster than using conventional methods. The Grid middleware provides dynamic coordination and interoperability for the heterogeneous objective functions in case of multi-objective optimisation. In addition, design experts have seamless access to evolutionary algorithms through the Grid portal which enhances their efficiency. There is also reliable security feature which allows legacy codes to be shared, yet ownership and intellectual property are preserved. Decision making for optimal solution is enhanced as different optimisation codes can be used for a

particular problem without much delay and results compared. An added advantage in using PSE for EC is the use of Grid visualisation instruments to display solutions in a graphical way. This aid understanding of the fitness functions.

The second scenario is in aerospace industry. Introducing a new aircraft into the market is a complex task that involves design experts who are hard to gather in one location. For example, Airbus manufactures different parts of its aircrafts in different European countries (Figure 5). The wings are manufactured in UK, the forward fuselage and vertical tail in Germany, the centre wing box and centre fuselage in France and the tail planes and central belly in Spain. To integrate these different designs of the different parts of the aircraft requires constant collaboration. Additionally, it requires huge computational power and knowledge sharing. The Grid offers a suitable environment for these design experts to share computational and optimisation algorithms to come up with a good product within the shortest possible time. Semantic and knowledge Grids can play a great role in capturing and reusing the design processes. Visualisation Grid which provides details for all experts involved in the project can aid in decision making and similar to the case of the automobile scenario, make the experts focus on achieving the same goal. The Grid middleware can overcome the heterogeneity of the different hardware and software used by the different experts. This Grid framework which allows experts to collaborate and share resources and services within the same company with different branches located in different countries is called Enterprise Grid [1].

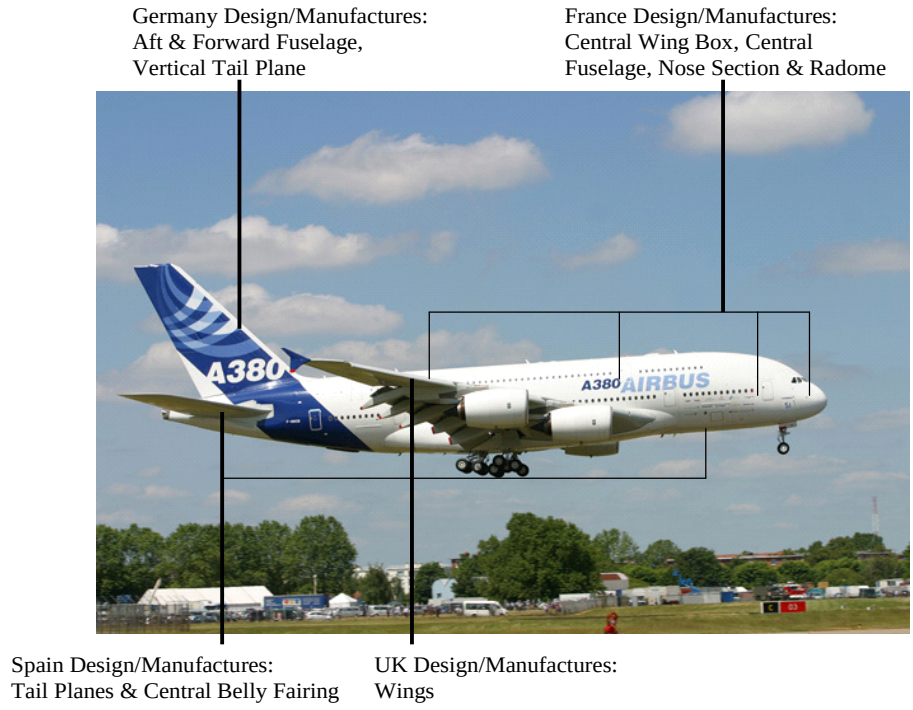


Figure 5 A380 with Different Parts Designed and Manufactured in Different

Countries (http://www.bbc.co.uk/southerncounties/community/airbus/parts_330.gif, 10/2/2007)

This aerospace scenario shows that the computations involved in the design of the different parts are distributed globally. Grid computing provides a platform upon which evolutionary computations can be subdivided and performed at the different nodes. Different algorithms can be used for the different aircraft parts for optimisation processes and be combined to form a hybrid evolutionary algorithm [14]. [14] used Hooke & Jeeves algorithm for local search and GA for evolutionary optimisation of heat processes and by combining the two algorithms (Hooke & Jeeves and GA), a more robust and efficient solution was obtained.

Computer Aided Design (CAD) systems are used for design of engineering parts within Grid-enabled environment. Evolutionary techniques are used as reverse-engineering mechanisms to produce CAD models so that the interactions of the interfaces of the various parts located at different Grid nodes and boundary conditions can be investigated under different constraints [30]. The Grid middleware can accommodate the heterogeneity in the CAD systems as well as the different objective functions used. Genetic algorithms usually guide the search for solutions using fixed fitness function but revisions to the criteria for the optimal solution involve manually modifying the fitness function [21]. Autonomic Grid computing can provide an agent-based evolutionary techniques that automatically changes the fitness function as the problem space and solution co-evolve in a dynamic manner [21].

Problem solving environments can be applied to all engineering applications because most other engineering disciplines require a range of experts. Therefore, the two scenarios are also applicable to many engineering products. It can be seen that PSEs integrate services, computational resources and people both within and outside organisations through Enterprise and Extraprise Grids. PSEs support remote problem definition that leads to the selection and application of appropriate engineering design search, exploration and optimisation techniques using evolutionary algorithms such as GA and genetic programming (GP) for the optimisation of multidisciplinary engineering processes [26]. The need for workflow management within this cross-domain and dynamic service demand and consumption calls for fuzzy timing techniques as part of the scheduling algorithms in PSEs [7]. Such workflow management strategies become more important when dealing with application areas that require high levels of precision in which case semantic modelling using Ontologies within the PSE is a good idea [4]. A combination of soft computing techniques such as neural networks and fuzzy logic termed as Neuro-Fuzzy gives an intelligent computational and optimisation feature to problem solving environments for complex design optimisation tasks [17]. It is important to note that different PSEs are designed for different optimisation applications because designers generally use approximation techniques to solve problems in order to reduce the otherwise enormous computational effort involved in each particular optimisation strategy [24]. In this regard specialised PSEs could solve either single-objective or multi-objective problems using different options of EC in each case.

Structural design based on parallel finite element analysis in engineering designs is usually parallelised for easy computation. Grid-enabled asynchronous genetic algorithm is parallelised to perform optimisation tasks at different nodes [25]. [25]

used Grid problem solving environment to parallelise evolutionary computing techniques parallel algorithms that serve as master-slave for the optimisation of simulation based designs to get better performance than conventional computing facilities offer. Grid computing focuses on large-scale resource allocation though its open service system. [9] used Grid-enabled platform to demonstrate how evolutionary computing resources can be allocated for global optimisation and search tasks. This system uses GA resources to maximise Grid resource allocation and reliability through dynamic security facility. Example of application areas that use Grid-enabled evolutionary computing techniques are engineering design optimisation, particle physics, bioinformatics, pharmaceutical and oil and gas.

4.3 Computational Steering and Visualisation Grid

Evolutionary computing techniques for multidisciplinary design optimisation which involves multi-objective functions may require the collaboration of distributed experts located in different geographical locations around the world. Computational steering allows each expert to make changes to particular parameters in the course of the optimisation to observe interesting patterns in the optimal solutions. Visualisation Grid located at the distributed nodes enhances the understanding and decision making during this process. Visualisation Grid can display the graphical representation of the fitness functions and convergence points during optimisation. Interrupting optimisation process for GA to observe patterns in the solutions using EC algorithms can help experts to pinpoint which parameter or parameters have great impact on the process. Advanced visualisation capabilities integrated with simulation based EC algorithms and design processes provide not only valuable design insight and design directions but also facilitate collaboration in design [19]. Close designer interaction with the design process supports exploration involving off-line processing and visualisation of initial results which leads to a redefinition of the solution space by dynamic evolutionary computing techniques [8]. [12] developed an online Grid visualisation system for data analysis. This system gives more meaning to the huge data produced by scientific instruments and evolutionary computing techniques. [11] described a multi-objective visualisation tool using Grid environment to represent engineering design information in multi-dimensional views. Commercial companies such as Boeing and Raytheon are using Grid-enabled Virtual Reality Network (VRN) for computational steering and visualisation on a real-time basis to improve their design processes [11]. Example of application areas that use computational steering are engineering design, physical sciences and life sciences.

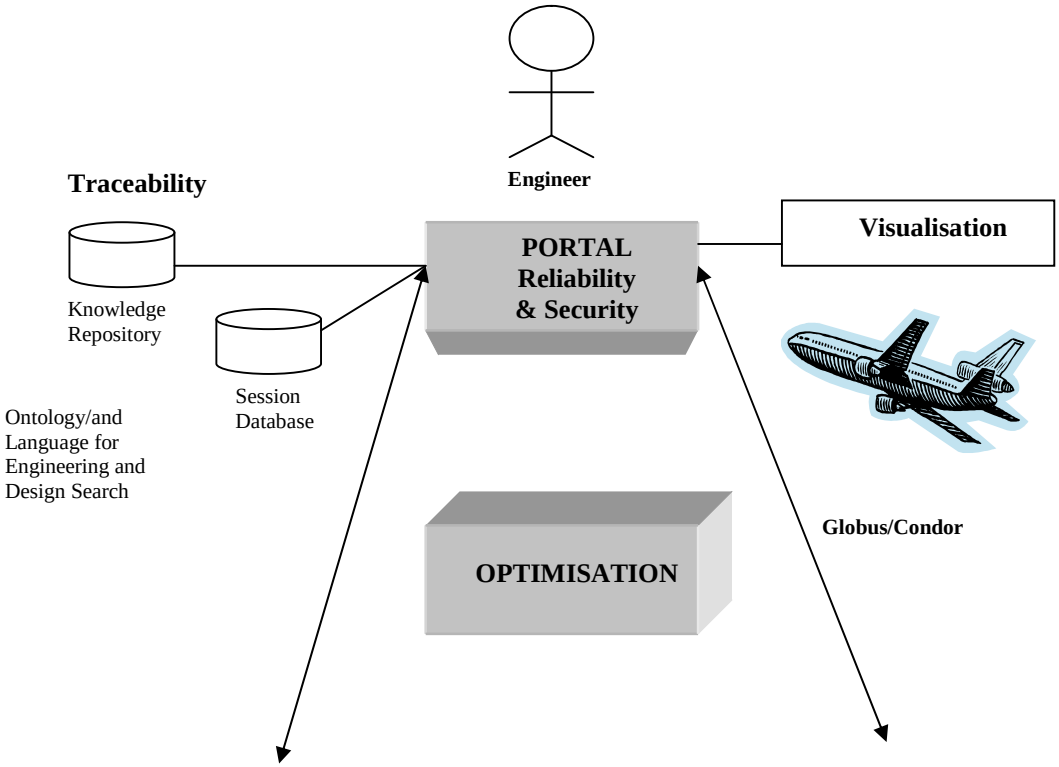
5 Examples of Intelligent Grid-Enabled Problem Solving Environments

There are Grid projects that have developed prototypes to demonstrate the use of optimisation techniques using EC algorithms and ontological intelligent systems. This section will describe briefly the working features of five of these projects. However,

there are many more of such projects that have been implemented within the Grid community. The projects described here are Geodise (Grid-Enabled Optimisation Design for Engineering), FIPER (Federated Intelligent Product Environment), SORCER (Service-Oriented Concurrent Environment), DAME (Distributed Aircraft Maintenance Environment) and Globus Toolkit (GT).

5.1 Geodise Project

Geodise is one of the numerous UK e-Science projects which aim to provide Grid-based seamless access to intelligent knowledge repositories, state-of-the-art collection of EC optimisation and search tools, analysis codes, distributed computing services and data resources (http://www.geodise.org/) for multidisciplinary engineering design optimisation. The project was funded by the UK EPSRC (Engineering and Physical Sciences Research Council) for an initial period of 3 years from 2001 to 2004 with a grant value of £2,872, 450 ($5,084,237). The project is based at the University of Southampton in UK with collaborating research partners at Universities of Oxford and Manchester also in UK. Collaborating companies are BAE Systems, Rolls Royce and Microsoft UK Ltd. The basic components of the Geodise Toolkit are application services, collaboration toolkits, data mining and analysis services, data management services, domain ontology and metadata, workflow services and problem solving environments. Geodise is usually used in conjunction with Matlab for its scripting and visualisation features. Matlab also have EC algorithms such as GA for optimisation and search applications. Geodise has proved to be suitable for engineering processes involving computational fluid dynamics (CFD). The use of Geodise tools by the engineer is facilitated by intelligent design advisors targeted initially at CFD [29]. Geodise is an open source software and is freely downloadable at the Geodise website. Figure 6 presents a simple Geodise architecture.



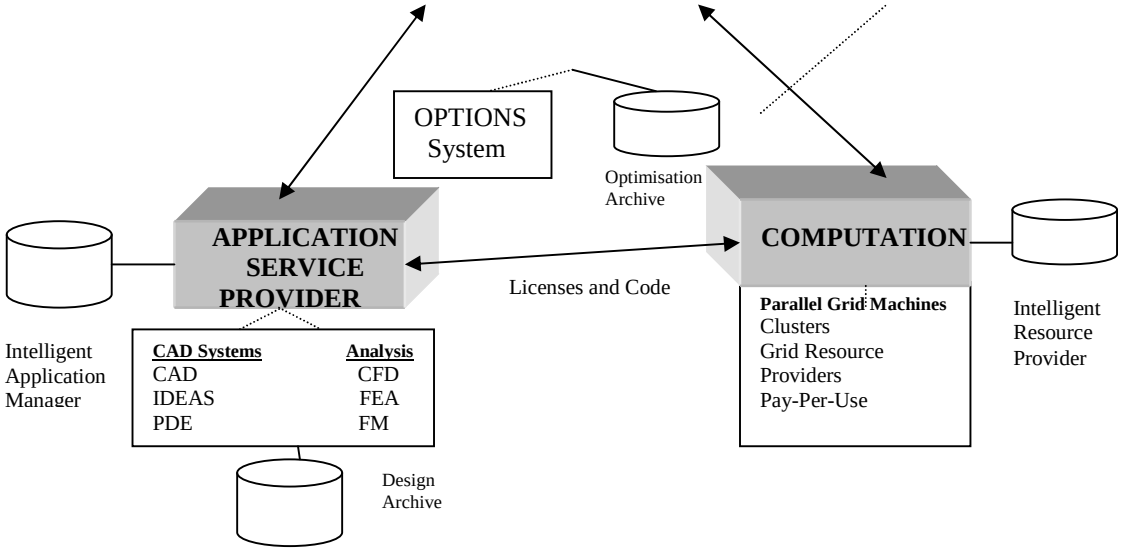


Figure 6 Simple Geodise Architecture

5.2 FIPER Project

FIPER is a 4-year project sponsored by the National Institute for Standards and Technology-Advanced Technology Program (NIST-ATP) in the U.S. Its aim is to produce an intelligent system that leverages the emerging web technologies in which engineering tools such as CAD (Computer Aided Design), CAE (Computer Aided Engineering), PDM (Product Data Management), EC optimisation techniques such as GA, GP and many others act as distributed service providers as well as service requestors communicating through intelligent context models for concurrent engineering design optimisation of FIPER has a grant worth of $21.5 million with Ohio and Stanford Universities as academic partners and GE (General Electric) teaming with Engineous Software, BFGoodrich, Parker Hannifin and Ohio Aerospace Institute as company partners [31]. FIPER uses GE Aircraft Engine functionalities to capture the designer's knowledge in Knowledge Based Engineering (KBE) systems to create Intelligent Master Model (IMM). This IMM contains the 'what', the 'why' and the 'how' of a design using EC algorithms to produce a range of global optimal solutions [31]. FIPER provides a graphical environment that permits interactive click-and-drag Grid programming interface which users can reuse and execute concurrently [35].

5.3 SORCER Project

SORCER is an extension of the work of FIPER project. SOCER lab is based at the Texas Tech University; U.S.A. The goal of SORCER is to form Grids of distributed

services that provide engineering data, applications, tools and EC optimisation algorithms for concurrent engineering design disciplines [37].

5.4 DAME Project

DAME is also a UK project which provides an intelligent Grid demonstrator system for health monitoring and fault trouble-shooting in mobile aircrafts. The need for mobile devices to have secured and seamless access to Grid services while actively working on the field is important in both scientific and business disciplines [23]. In this framework, EC algorithms are used to select optimal diagnostic solutions based on certain constraints and parameters using multiple objective functions for the optimisation.

5.5 Globus Tookit

Globus is an open source toolkit considered as the de facto middleware for large-scale computing services. The Globus toolkit is funded by various organisations such as the Defense Advanced Research Projects Agency (DARPA) USA, National Science Foundation (NSF) USA, Globus Alliance and the UK e-Science program. The I-WAY (Information Wide Area Year) projects which succeeded in linking many supercomputing centres was the first initiative that led to the development of the Globus toolkit in 1995. Globus toolkit consists of many services and protocols such as Globus Security Infrastructure (GSI), Globus Information Services (GIS) and Globus Resource Allocation and Management (GRAM). Globus has evolved to the current version 4.0 with many additions to the protocols DAI (Data Access and Integration) developed by the UK e-Science Centre and the standard 'plumbing' features of the OGSA (Open Grid Services Architecture) which provides standard implementation interface for participants (service providers and service requestors) in the Grid so as to ease and facilitate interoperability among heterogeneous resources and services. To join in the Grid, local users must follow the standard protocols to reduce customisation efforts. This philosophy encourages autonomic Grid resource management. This autonomic characteristic of the Globus toolkit is where the use of EC algorithms is useful for dynamic adjustment and adaptation strategies for resource management. Adaptation is the key feature that will make the complex Grid composition a success. This will help software agents to dynamically change states to suit their environments. Multi-objective optimisation will play a great role in scheduling resources and services for requestors through efficient assignment of the resource to the right users based on inference rules within evolutionary context.

6 Collaborative Evolutionary Grid Computing

The original intention of the Grid is to link supercomputing stations for collaboration. The sharing and dynamic coordination of computational resources involves policies that govern the activities of service providers and services requestors. A set of

individuals and institutions that define rules about what should be accessed and who should access them is called virtual organisation. These types of Grids which share resources and services for common scientific or business goals are called collaboratories [28]. The field of evolutionary computing needs collaboratories to explore more evolutionary algorithms that can tackle more complex optimisation problems. Grid-enabled collaboratories will enhance distributed and secured access to computational power, data management resources and EC algorithms. Real-time solution for evolutionary computations is guaranteed due to the synergy of distributed computational resources within the virtual organizations that made up a collaboratory.

2 Summary and Conclusions

This chapter presented how Grid Computing can enhance the functionalities provided by EC techniques through its distributed and dynamically coordinated computational resources and services. Grid resources include hardware, software, instruments, visualisation systems and evolutionary algorithms. Grid-based architectures such as service-oriented architecture (SOA) and problem solving environments (PSEs) were discussed. Examples of intelligent PSEs such as Geodise, FIPER, SORCE, DAME and Globus were briefly explained within the context of evolutionary optimisation applications.

Grid has become a key platform upon which large-scale computations can be performed in a more coordinated way. Evolutionary computing (EC) is suited for Grid application due to its computational and data intensive nature. EC algorithms produce enormous solutions for complex multi-objective optimisation problems and the Grid, using its distributed nodes, can handle the selection of an optimal solution from the feasible set of solutions. Evolutionary computations of multiple objective functions can be physically carried out in different locations around the world and the results converged at the point of need. Data Grid has capabilities for data access and integration of multiple data produced by multiple objective functions during computations. The heterogeneity in different objective functions and constraints can be overcome using Grid middleware during computations done at different locations. In addition, the middleware allows different software and hardware to interoperate which enhances the use of different software for different objective functions. In addition, collaborative evolutionary computing framework within Grid environment can enhance new discoveries in evolutionary computation. Grid collaboratories can allow different experts in optimization processes located at different places to work together and share their expertise. Optimisation processes using evolutionary algorithms are done under confidential environments by companies due to the competitive nature of the economy and the need to safeguard the intellectual property of the company. Grid security feature which has single sign-on proxy encryption protocols provide secured platform for this multidisciplinary collaboration. Semantic and Knowledge Grids provide intelligent problem solving environments for design engineers who can have easy access to computational codes such as GAs. It is important to note that the use of EC techniques within Grid environment is still at its

infancy and much work needs to be done in this area. However, Grid promises to bring revolutionary efficiency in the way EC algorithms are used.

References

1. Abbas, Ahmar. Grid Computing: A Practical Guide to Technology and Applications. USA: Charles River Media, Inc. (2004).

2. Agarwal, Manish and Parashar, Manish. Enabling Autonomic Compositions in Grid Environments. Proceedings of the Fourth International Workshop on Grid Computing (GRID'03). USA: IEEE Computer Society. (2003).

3. Back, Thomas; Hammel, Ulrich, and Schwefel, Hans-Paul. Evolutionary Computation: Comments on the History and Current State. IEEE Transactions on Evolutionary Computation. (1997) 13-17.

4. Cannataro, Mario; Comito, Carmela, and Guzzo, Antonella Veltri Pierangelo. Integrating Ontology and Workflow in PROTEUS, a Grid-Based Problem Solving Environment for Bioinformatics. Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04). USA: IEEE Computer Society. (2004).

5. Cannataro, Mario and Talia, Domenico. Semantics and Knowledge Grids: Building the Next-Generation Grid. IEEE Intelligent Systems. (2004) 56-63.

6. Cannataro, Mario and Talia, Domenico. Towards the Next-Generation Grid: A Pervasive Environment for Knowledge-Based Computing. Proceedings of the International Conference on Information Technology: Computers and Communications (ITCC'03). USA: IEEE Computer Society. (2003).

7. Cao, Junwei; Jarvis, Stephen A., and Saini, Subhash Nudd Graham R. GridFlow: Workflow Management for Grid Computing. Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03). USA: IEEE Computer Society. (2003).

8. Cvetkovic, Dragan and Parmee, Ian. AgentBased Support within an Interactive Evolutionary Design System. Parmee, I. C. Adaptive Computing in Design and Manufacture. 5th ed. London, UK: Springer-Verlag London Limited, (2002): 355-367.

9. Dai, Yuan-Shun and Wang, Xiao-Shun. Optimal Resource Allocation on Grid Systems for Maximizing Service Relaibility using a Genetic Algorithm. Elsevier: Reliability Engineering and System Safety. (2005).

10. Deb, Kalyanmoy. Multi-Objective Optimisation using Evolutionary Algorithms. England, UK: John Wiley & Sons Ltd. (2001).

11. Eddy, John and Lewis, Kemper. Multidimensional Design Visualization in Multiobjective Optimization. 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization . USA: AIAA Inc. (2002).

12. Foster, Ian; Insley, Joseph; Laszewski, Gregor von; Kesselman, Carl, and Thiebaux, Marcus. Distance Visualization: Data Exploration on the Grid. IEEE Computer Society. (1999) 26-43. CODEN: 0018-9162/99.

13. Foster, Ian and Kesselman, Carl. The Grid: Blueprint for a New Computer Infrastructure. San Francisco, USA: Morgan Kaufman Publishers Inc. (1999).

14. Fraga, E. S. and Zilinskas, A. Experience with Hybrid Evolutionary/Local Optimization for Process Design . Parmee, I. C. Adaptive Computing in Design and Manufacture. 5th ed. London, UK: Springer -Verlag London Limited. (2002) 53-64.

15. Gil, Yolanda, Deelman, Ewa, Blythe, Jim, Kesselman, Carl and Tangmunarunkit, Hongsuda. Artificial Intelligence and Grids: Workflow Planning and Beyond. IEEE Intelligent Systems. (2004) 26-33.

16. Goux, Jean-Pierre, Kulkarni, Sanjeev, Yoder, Michael and Linderoth, Jeff. Master-Worker: An Enabling Framework for Applications on the Computational Grid. Kluwer Academic Publishers: Cluster Computing. (2001) 463-70.

17. Joshi, Anupam, Weerawarana, Sanjiva, Ramakrishnan, Narendran, Houstis, Elias N. and Rice, John R. Neuro-Fuzzy Support for Problem-Solving Environments: A Step Toward Automated Solution of PDEs. IEEE Computational Science & Engineering: Neural Networks Problem Solving Environments. (1996) 44-56.

18. Kacsuk, P., Goyeneche, A., Delaitre, T.; Kiss, T., Farkas, Z. and Bocko, T. High-Level Grid Application Environment to Use Legacy Codes as OGSA Grid Services. Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (GRID'04). USA: IEEE Computer Society. (2004).

19. Kodiyalam, Srinivas; Yang, R. J., and Gu, Lei. High Performance Computing & Rapid Visualization for Design Steering in MDO. 44th AIAA/ASME/ASCE/AHS Structures, Structural Dynamics, and Materials Conference. USA : AIAA Inc. (2003).

20. Li, Yuhua and Lu, Zhengding. Ontology-Based Universal Knowledge Grid: Enabling Knowledge Discovery and Integration on the Grid. Proceedings of the 2004 IEEE International Conference on Services Computing (SCC'04). USA: IEEE Computer Society. (2004).

21. Masher, M. L. and Garza, A. Gomez de Silva. Adapting Problem Specifications and Design Solutions Using Co-evolution. Parmee, I. C. Adaptive Computing in Design and Manufacture. 5th ed. London, UK: Springer-Verlag London Limited. (2002) 257-271.

22. Naik, Vijay K.; Sivasubramanian, Swaminathan, and Krishnan, Sriram. Adaptive Resource Sharing in a Web Services Environment. IFIP International Federation for Information Processing. (2004) 311-330.

23. Ong, M.; Alkarouri, M.; Ren, X.; Allan, G.; Kadirkamanathan, V.; Thompson, H. A., and Fleming, P. J. Grid-Based Decision Support with Pro-Active Mobile Computing. Proceedings of the 2005 IEEE International Conference on Services Computing (SCC'05). USA: IEEE Computer Society. (2005).

24. Papazoglou, Mike P. Service-Oriented Computing: Concepts, Characteristics and Directions. Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE'03). USA: IEEE Computer Society. (2003).

25. Park, Jin Woo, Park, Si Hyoung, Moon, Ji Joong, Yoon, Youngha and Kim, Seung Jo. High-Fidelity SimulationBased Optimum Design Utilizing Computing Grid Technology. 46th AIAA/ASME/ASCE//AHS/ASC Structures, Structural Dynamics & Materials Conference. USA: AIAA Inc. (2005).

26. Parmee, I. C., Abraham, J., Shackelford, M., Rana, O. F. and Shaikhali, A. Towards Autonomous Evolutionary Design Systems via Grid-Based Technologies. Proceedings of ASCE 2005 International Conference on Computing in Civil Engineering (IPDPS'03). ASCE. (2005).

27. Pattnaik, Pratap, Ekanadham, Kattamuri and Jann, Joefon. Autonomic Computing and Grid. Berman, Fran, Fox, Geofrey C. and Hey, Anthony J. G. Grid Computing: Making the Global Infrastructure a Reality. England, UK: Jonh Wiley & Sons Ltd. (2003) 351-384.

28. Pearl, Laura , Welch, Von, Foster, Ian, Carl, Kesselman and Tuecke, Steven. A Community Authorization Service for Group Collaboration. Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks (POLICY'02). USA: IEEE Computer Society. (2002).

29. Pound, G. E. , Eres, M. H., Wason, J. L., Jiao, Z., Keane, A. J. and Cox, S. J. A Grid-Enabled Problem Solving Environment (PSE) for Design Optimisation

within Matlab. Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03). USA: IEEE Computer Society. (2003).

30. Robertson, C. and Fisher, R. B. Better Surface Interactions by Constrained Evolution. Parmee, I. C. Adaptive Computing in Design and Manufacture. 5th ed. London, UK: Springer-Verlag London Limited. (2002) 133-142.

31. Rohl, Peter J.; Kolonay, Raymond M.; Irani, Rohinton K.; Sobolewski, Michael; Kao, Kevin, and Bailey, Michael W. A Federated Intelligent Product Environment. AIAA. (2000).

32. Roure, David De; Jennings, Nicholas R., and Shadbolt, Nigel R. The Semantic Grid: A Future e-Science Infrastructure. Berman, Fran//Fox, Geofrey C.//Hey, Anthony J. G. Grid Computing: Making the Global Infrastructure a Reality. England, UK: John Wiley & Sons Ltd. (2003) 437-470.

33. Schikuta, Erich and Weishaupl, Thomas. N2Grid: Neural Networks in the Grid. IEEE. (2004) 1409-1414.

34. Scurr, A. D. and Keane, A. J. The Development of a Grid-Based Engineering Design Problem Solving Environment. Parmee, I. C. Adaptive Computing in Design and Manufacture. 5th ed. England, UK: Springer-Verlag London Limited. (2002) 65-73.

35. Sobolewski, Michael and Kolonay, Raymond M. Federated Grid Computing with Interactive Service-Oriented Programming. Concuurent Engineering: Research and Applications. (2006) 1455-66.

36. Song, Wenbin ; Ong, Yew Soon; Ng, Hee Khiang; Keane, Andy; Cox, Simon, and Lee, Bu Sung. A Service-Oriented Approach for Aerodynamic Shape Optimisation Across Institutional Boudaries. 2004 8th International Conference on Control, Automation, Robotics and Vision. USA: IEEE. (2004).

37. Soorianayanan, Sekhar and Sobolewski, Michael. Monitoring Federated Services in CE Grids. Sobolewski, Michael//Cha, Jianzhong. Concurrent Engineering: The Worldwide Engineering Grid. China: Tsinghua University Press and Springer-Verag. (2004) 89-95.

38. Wager, Tor D. and Nichols, Thomas E. Optimisation of Experimental Design in fMRI: A General Framework using a Genetic Algorithm. IEEE NeuroImage Academic Press. (2003) 18293-309.



https://dspace.org/

Evolutionary Computing within Grid Environment.

Tiwari, Ashutosh; Goteng, Gokop; Roy, Rajkumar

2007-01-01T00:00:00Z

http://dx.doi.org/http://www.springer.com

Ashutosh Tiwari, Gokop Goteng and Rajkumar Roy, Evolutionary Computing within Grid Environment. Studies in Computational Intelligence, 66, pg. 229-248

Downloaded from a DSpace repository. https://dspace.org/