Proceedings of the 5th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed Systems, Madrid, Spain, February 15-17, 2006 (pp43-47)

An Application of Genetic Algorithm for Auto-body Panel Die-design Case Library Based on Grid

Demin Wang[2], Hong Zhu[1], and Xin Liu[2]

[1]College of Computer Science and Technology, Jilin University,
Changchun 130012, P.R. China
[2]The Network Center, Jilin University,
Changchun 130021, P.R. China

Abstract: - The use of case library techniques to direct new designs for auto industry is becoming increasing widespread. In this paper, we use a genetic algorithm to realize a quick match with the best case, which can satisfy the intelligence request of the case library. Experiment results show that the proposed algorithm can greatly increase the efficiency of searching for the best case. Additionally, this paper presents a novel framework of auto-body panel die-design case library based on grid. By using grid technology, it has solved the principal problems existing in current case libraries. Consequently, we anticipate that this work will have a broad application in the making of auto-body panel dies.

Key-Words: - Genetic Algorithm**,** Grid, Case Library, Auto-body panel, die

1 Introduction

With the rapid development of auto industry, auto styles are updating gradually, and the period of updating auto styles between two generations is becoming shorter and shorter. As such, the conventional designing and producing methods of dies cannot meet the need of rapid development any more. Although experience plays an important role in auto die design, training a better designer will cost a lot of time. Nowadays, those die-design people with much hands-on experience have been laid off or left those enterprises, which cause great loss of such knowledge. Hence, if we are able to retain the design experience, later designers can design a new scheme by combining and revising those old designing schemes. This will have a broad application in the making of auto-body panel dies. Based on the complex and dynamic behaviors of case library, this paper presents an organizing scheme by incorporating mobile-body panel die-design case library. It better shows the relation between standard pieceworks and dies, and realizes the quick retrieval of pieceworks for designers to use.

Case library intelligence is a certain trend in the development of the case library. The pith is concerned with transforming information into knowledge and then directing the originality by using this knowledge. At present, the best case retrieving methods in widespread use is to consider the characteristic values input by the user as the present case, and then searching for the most similar case with the present case in case libraries in order to direct the new designs for users. However, case libraries are gradually enlarged with the accumulation of applications, so how to efficiently retrieve the objective case has become the key problem. In the past, a sequential matching method is widely used in the database searching. But this will be very boring and ineffective because the auto-body panel die-design case library has a multiple hierarchical structure and a huge amount of complex data. In this paper, we use a genetic algorithm, and write the relevant characteristic values into a file, which can be read into the memory once. Experiment results show that the proposed method can greatly increase the efficiency of retrieving the best case.

Additionally, the sharing of case libraries has not been implemented to any large extent. Three main problems are shown as follows. Firstly, most of suppliers and producers live in different countries and regions, so the information platform has not been formed yet and thus the applications of the case library are restricted to small enterprises; Secondly, communication among enterprises is very limited; Finally, heterogeneous case libraries dispersed in different geographical regions are difficult to realize the true share. Grid is viewed as a new important infrastructure, however, it has more significant features than other network technologies, and its

main characters are that it can implement sufficient share for these geographically distributed resources. It also can remove the difference of platforms and realizations among all nodes. This paper proposes a framework for auto-body panel die-design case library based on grid. Constructing grid-based case libraries has shown the prospect of applying grid technology to the case library, and thus has a bigger theoretical meaning and practical application value.

2 Genetic Algorithm

The concept of genetic algorithm was first presented in 1975 by John Holland [1]. It is a stochastic search method which is inspired by evolution in biological systems where the search is conducted directly in the solution space. Each solution is encoded in a certain way and is called an *individual*. The search is parallel in the sense that a *population* of individual is maintained and the quality of the individuals is calculated by a *fitness function.* The population is improved by crossover, recombination of genetic material from different individuals. This is based on a hypothesis that a good solution can be built up from shorter partial solutions [1-3]. Genetic diversity is maintained by a mutation operation, making random changes in the individuals.

When the genetic algorithm is implemented it is usually done in a manner that involves the following cycle: Evaluate the fitness of all of the individuals in the population. Create a new population by performing operations such as crossover, fitness-proportionate reproduction and mutation on the individuals whose fitness has just been measured. Discard the old population and iterate the new population. Generally, genetic algorithm is described as follows:

1. Generate the Initial Population;
2. Evaluate the Population;

While No termination criterion is satisfied **do**

3. Select chromosomes from the current population;
4. Apply the Crossover and Mutation operators to the chromosomes selected at step 1 to generate new ones;
5. Evaluate the chromosomes generated at step 4;
6. Apply the Acceptation criterion to the set of chromosomes selected at step 3, together with the chromosomes generated at step 4;

End while

7. Return the best chromosomes evaluated so far;

3 Auto-body Panel Die-design Case Library Based on Grid

3.1 Organization of the Case Library

There are many ways to realize auto-body panel die-design case library [4]. We choose a relational database. The realization of the case library is indexed by parts and it is classified into directory structure and case data, which is shown as follows:

1. Case logical directory structure is divided into five layers, Automobile classification, vehicle type, a series of specific types, components assembly type, and the name of some certain workpieces, which are mapped to some interrelated tables in relational database.
2. Case data is divided into two layers: The first contains concrete case data of workpieces and the second contains concrete case data of some forming process dies belonging to this workpiece. The case data of case library consists of these two layers, which are mapped to some interrelated tables in relational database.

In order to search for the best case to direct the new design for users more conveniently, we draw some main characteristic values of each part or die when we add new cases each time. Then write these features into a certain file and store this file into the database as a single field.

3.2 Framework

According to the distance to the shared resource, a hierarchical model is presented. This model, as figure1 shows, provides a hierarchical abstraction of the case library based on grid. We begin the discussion by understanding each of the layers in the model.

At the lowest level, the information layer, we have the case libraries that Grid users want to share and access. Moreover, the case libraries are distributed on different platforms, and they store the data by using different database systems.

The next horizontal layer, the interactive layer, consists of the information interactive layer and the service interactive layer built atop the information layer. The information interactive layer contains a bunch of GCSs (Grid Case Library Service). They are provided to the upper level as grid services, and developed by the administrator or the users of each node. The service interactive layer can be viewed as a collective center of services. It contains CNMS (Case library Node Management Service) and GCSR (Grid Case library Service Registry). GCS receives the administration and status monitoring of CNMS of the local case library nodes, and when a GCS has changed in this case library, it should send a message

to GCSR in order to register a new service or cancel the original one. GCSR is responsible for processing the registration messages which are sent from CNMS, and storing these messages into the registration library. GCSR will also send detecting messages at a certain interval to CNMS, and get the lastest status information of all nodes in order to provide more efficient service discovery for CLAS (Case Library Aggregation Service).

Above the interactive layer is the collective layer. It contains CLAS, which directly receives the request from the user who has an approved identity validation, and then retrieves the relevant factory service from GCSR. Finally, CLAS gets the result and processes it to the needed format.

At the top of this model are user applications. They can get results from the collective layer and control the format of displaying cases.

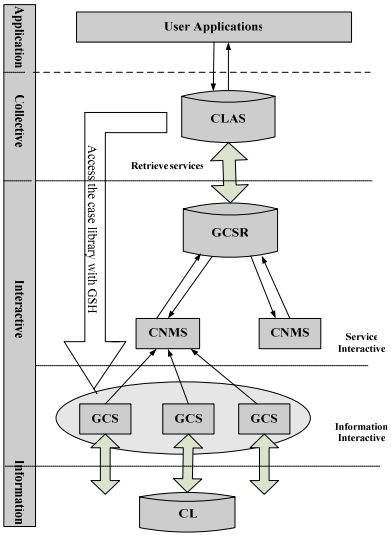

Fig.1. The Architecture of the Case Library Based on Grid

4 Genetic Algorithm for Grid-based Auto-body Panel Die-design Case Library

4.1 Encoding

Every record is read as an element into a vector. If the amount of the records is N, number them from 1 to N. Here we set m is the length of binary string N, then every individual in the population adopts the unsigned binary code {0,1} with the same length m, referring to the location of every record. For example, if N=5000, then m=13 and the 1000th record is encoded with 0001111101000.

4.2 Fitness Function

According to the strategy of near neighbor, fitness function is defined as follows:

$$Similarity(T,S)=\sum_{i=1}^{n} f(Ti,Si)\times Wi \qquad (1)$$

Where T refers to the target case and S is the source case; n is the number of similarity units contained in each case. $f(Ti,Si)$ is defined as the similar value of the i-th similarity unit; Here similarity unit is the similar element of two parts, which make a similar comparison [5]. We take some oil pan as an example, shown in the figure 2, it contains a base similarity unit a, oil seal similarity units b and c, and step similarity units d and e. Wi represents the weight of each similarity unit which is input by the client, $Wi\in[0,1]$, $\sum_{i=0}^{n} Wi=1$. $i\in[1,n]$.

In addition, $f(Ti,Si)$ is defined in Equation (2). $f(Ti,Si)_n$ is the fitness of the number of characteristic values and $f(Ti,Si)_s$ refers to the fitness of the characteristic values of a similarity unit; n_s and n_t are the amount of characteristic values of a similarity unit in the source case and the current case respectively, whose values are not equal to –1; -1 indicates no values; m is the number of the characteristic values of a similarity unit; r_{js} and r_{jt} are the j_{th} characteristic value of a similarity unit in the source case and the current case dividedly.

$$f(Ti,Si)=f(Ti,Si)_n\times f(Ti,Si)_s$$

$$f(Ti,Si)_n=\begin{cases} n_t/n_s, n_t\le n_s \\ n_s/n_t, n_t> n_s \end{cases} \qquad (2)$$

$$f(Ti,Si)_s=\begin{cases} \frac{1}{m}\sum_{j=1}^{m}\frac{r_{js}}{r_{jt}}, r_{js}\le r_{jt} \\ \frac{1}{m}\sum_{j=1}^{m}\frac{r_{jt}}{r_{js}}, r_{js}> r_{jt} \end{cases}$$

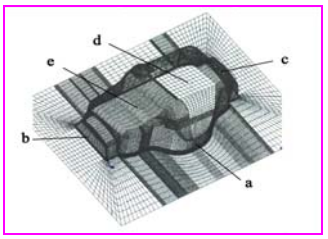

Fig.2. The similarity units of an oil pan.

4.3 Selection

In order to avoid premature convergence, we perform a sigma proportional transformation on each individual's fitness value [6]. For example, for the fitness value f(i) of the i-th individual, at first we

apply the following formula to f(i) to transform it into ExpVal(i):

$$ExpVal(i) = \begin{cases} 1+(f(i)-f(t))/2\sigma(t), \sigma(t) > 0 \\ 1, \sigma(t) = 0 \end{cases} \quad (2)$$

Where f(t) is the average fitness value of the t-th generation population, and $\sigma(t)$ is the standard deviation of the t-th generation population. After such transformation, the algorithm then uses elitist fitness proportionate selection mechanism for ExpVal(i) to select chromosomes for reproduction. The best individual in the population is always passed on unchanged to the next generation, without undergoing crossover or mutation.

4.4 Crossover

The crossover probability P_c is set to 0.75 against this problem and the crossover operator is defined as follows: The two-point crossover generates two new individuals by randomly selecting two cross sites and exchanging the corresponding coordinates between them. If the number of any new individual is greater than the amount of the records, discard these two individuals and do the crossover operation again to get a new couple.

4.5 Mutation

The mutation operators are applied sequential and independently from crossover. It enables the algorithm to have the ability to local search and keep the diversity of the population. The gene is mutated stochastically with a given mutation rate P_m. If the gene is '1', we set it to '0', otherwise, if the gene is '0', we set it to '1'. And if the number of any new individual is greater than the amount of the records, discard this individual and do the mutation operation again to get a new one.

5 Experimental Results and Analysis

The algorithm described above was implemented as a grid service. It is developed to each node of the case library using the bottom-up method [7]. In order to verify the efficiency and validity of the genetic algorithm, 5000 and 10000 records of simulated data related with some oil pan are produced and two experiments are performed respectively. The experimental parameter values are shown in table 1. Table 2 shows the comparison of different methods. As can be seen, the GA method is about 23 times faster than the traditional method ATDB (accesses the database item by item) against a 5000 records dataset and nearly 25 times regarding a dataset containing 10000 records. These results show that the GA method is considerably more effective than traditional searching methods. It can be concluded that in practical applications, the GA method can increase the usefulness of CPU by increasing its efficiency in searching.

Table 1. The parameter values of the experiment

Parameters	Parameter values	
	5000 records	10000 records
Population size	20	30
Generation count	100	150
Crossover probability	0.50	0.75
Mutation probability	0.01	0.03

Table 2. Comparison of different methods for searching the best case in the grid-based case library

Methods / Number of records	CPU Time (s)			
	ATDB		GA	
	Min.	**Ave.**	**Min.**	Ave.
5000	**48.93**	**58.08**	**2.01**	2.48
10000	102.42	121.43	4.52	4.86

Note: ATDB is a traditional method that accesses the database item by item; GA is genetic algorithm; Min. is minimum CPU time in twenty operations; Ave. is average CPU time in twenty operations.

Figure 3 provides a snapshot of the grid-based case library client tool. The interface displays the directory structures of the case library, and provides the query function of searching the case library data by keywords.

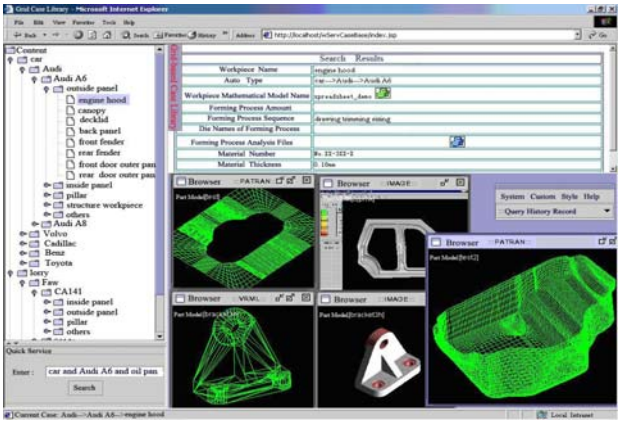

Fig.3. Grid-based case library client tool

6 Conclusion and Future Work

In this paper, we present a framework of grid-based case library by incorporating mobile-body panel die-design case library, which uses grid technology to implement virtual integration of heterogeneous case libraries dispersed in different geographical

regions, and such case library has solved the principal problems existing in current case libraries and enhances collaboration and communication among different enterprises. In addition, we use a genetic algorithm to realize the quick match with the best case, which can satisfy the intelligence request of the case library. Experiment results show that the proposed algorithm can greatly increase the efficiency of searching for the best case, which will have a deeper meaning for the application of the case library under the grid environment.

The core of next work will combine with more knowledge of optimization algorithms to make sure that the retrieved case by our method has a higher similarity accuracy compared with the objective case.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 60433020.

References:

[1] Holland, J. H., *Adaptation in Natural and Artificial Systems*, MIT Press, USA (1992).

[2] Davis, L., *Genetic Algorithm and Simulated Annealing*, Pitman Publishing, UK (1987).

[3] Goldberg, D.E., *Genetic Algorithm in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company, USA(1989).

[4] Wang Demin, Wu Zhiqiang and Hu Ping, Key techniques to distributed die design case library, *Journal of Jilin University (Engineering and Technology Edition)*, vol. 35, pp. 314-318, May 2005.

[5] Zhou Meili, Principles of Similarity Formation between Similar Systems, *Int. J. of General System*, vol. 27(1), pp. 495-504, 1999.

[6] Eloranta, T., Makinen, E., *TimGA—a Genetic Algorithm for Drawing Undirected Graphs*. Technical report, Department of Computer Science, University of Tampere, December(1996).

[7] Borja Sotomayor . (2003). *The Globus Toolkit 3 Programmer's Tutorial*. [Online]. Available: http://gdp.globus.org/gt3-tutorial/.