International Journal of Production Research,
Vol. 46, No. 21, 1 November 2008, 6113–6135

Designing data warehouses for equipment management system with genetic algorithms

K.-Y. CHEN*†, M.-C. CHEN‡ and W.-Y. LIU§

†Department of Industrial Engineering and Management, National Taipei University of Technology, Taiwan, ROC. No. 1, Section 3, Chung-Hsiao E. Road, Box 2337, Taipei 106, Taiwan ROC
‡Institute of Traffic and Transportation, National Chiao Tung University, Taiwan, ROC
§Department of Industrial Engineering and Management, National Taipei University of Technology, Taiwan, ROC

(*Revision received December 2006*)

Effective equipment management is becoming one of the key factors in keeping a competitive advantage in the dynamic business environment since equipment is an important asset for manufacturing companies. Nowadays, maintenance administration has become one of the most important tasks in equipment management, particularly in manufacturing industries. Equipment management system (EMS) aims at reducing maintenance cost and production loss caused by machine breakdown. In addition, EMS can assist equipment engineers to make the right maintenance decisions at the right time, and at the right shop floor. Traditional computerized maintenance management systems (CMMS) have helped equipment engineers to deal with maintenance operations, but they lack decision support capability. In this paper, we design a data warehouse (DW) for EMS to help equipment engineers make maintenance decisions with various equipment related dimensions to improve effectiveness. A set of cubes can be built from EMS DW for the purpose of decision-making. In order to achieve a reasonable query response time under the memory space limit, a mechanism of partial materialization based on genetic algorithms (GAs) is adopted to design data cubes in the EMS DW. From the computational results the proposed GA-based approach for cube design can be applied to effectively select the appropriate multi-dimensional views for equipment management.

Keywords: Equipment management system; Maintenance management; Data warehouse; Genetic algorithms

1. Introduction

In recent years advanced manufacturing systems and technologies have continued to gain interest in the increasingly competitive business environment. Enterprises have realized that engineering alone is insufficient for an effective manufacturing system; an advanced management system is essential as well. In a highly automated manufacturing environment, equipment is an important asset since its conditions seriously affect the production capacity, product quality and process yield.

*Corresponding author. Email: kychen@ntut.edu.tw

International Journal of Production Research
ISSN 0020–7543 print/ISSN 1366–588X online © 2008 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/00207540701222776

As a result, effective equipment management becomes one of the essential tasks for enterprises in order to keep their competitive advantage in the dynamic business environment.

As information and Internet technologies have become more accessible and powerful, enterprise data and information have dramatically increased and have become more vital in decision-making. These rapid advancements accelerate the massive information flow in intra-enterprise and supply chains. With e-manufacturing the manufacturing information can be incorporated with enterprise information systems (EIS) to support decision-making for business management. Constructing data warehouses (DWs) for supporting decision-making is taken as the underpinning of e-manufacturing. Equipment management system (EMS) with e-maintenance and e-diagnostics effectively support the equipment monitoring and control, efficiency analysis and failure prediction. In EMS, DW integrates data and information from a large set of manufacturing facilities to support e-maintenance and e-diagnostics. Extracting data directly from one or more operational and transactional databases provides an opportunity for aggregating data and information for decision-making purposes.

Equipment management involves decision-making on what work is to be done, when the work should be done, how long it should take, what materials, labour skills and tools are needed and are available. Proper decisions require timely, coherent and accurate data in terms of many factors such as system configuration, equipment operation and repair/maintenance history, cost, availability and requirement of resources, the production schedule, spare parts inventory, etc. However, some of these data may not be available in the maintenance department but may be accessed through a distributed information system. As the scale and complexity of both the manufacturing system and the equipment management system expands, it is obvious that all the relevant information cannot be effectively managed without a well-designed database.

EMS aims at reducing equipment maintenance cost and production loss caused by machine breakdowns while assisting equipment engineers to make the right maintenance decisions at the right time, and at the right shop floor. The decision support system can be applied to assist diagnosis, maintenance planning, and scheduling of manufacturing systems (Jeong *et al.* 2006). The real-time equipment monitoring and control, data collection and failure analysis are achieved through e-maintenance, which attempts to realize the objective of near-zero downtime in manufacturing systems (Raman 2001, Intelligent Maintenance Systems 2005, Hung *et al.* 2005). Equipment engineers can repeatedly acquire massive amounts of equipment data from the automated manufacturing system. However, the detailed and tedious operational data may not be directly used to support decision-making for equipment management. The breakdown of this data into succinct useable units is required. Provided that these data are adequately collected and explored, EMS can improve the productivity by reducing equipment breakdown, increasing spare part availability and optimizing resource allocation.

In this paper, we design a DW for EMS to help equipment engineers make maintenance decisions with various equipment related dimensions in an effective manner. Several cubes such as 'mean time to repair' (MTTR), 'mean time between failure' (MTBF), 'spare part response time' (SPRT), can be built from the EMS DW for the purpose of decision-making. In order to achieve a reasonable query response time under the memory space limit, a data warehousing strategy of partial materialization is adopted to design data cubes. With the DW design it is extremely difficult to obtain optimal solutions for large-scale problems. This paper applies genetic algorithms (GAs) (Holland 1975, Goldberg 1989) to deal with DW design problems. GAs were successfully used to resolve various complicated optimization problems in manufacturing (e.g. Chen 2000, Chiang and Su 2003, Hsu *et al*. 2005, Wu *et al*. 2006, Su and Chiang 2002, Su and Shiue 2003). Some recent review papers reported the growing GA applications in many production and operations management problems such as production scheduling, facility layout, production planning, supply chain design, process condition optimization, etc. (Aytug *et al*. 2003, Chaudhry and Luo 2005, Oduguwa *et al*. 2005). In this paper, an approach hybridising GA and greedy search is used to optimally materialize the EMS DW for the semiconductor packaging manufacturing. Additionally, we compare the DW designs proposed by our approach to that of other alternative methods in the literature. The remainder of this paper is organized as follows. Section 2 presents a review of the DW design literature. Section 3 introduces the architecture of EMS DW. The mathematical model of cube selection for DW materialization is given in section 4. The GA-based approach for optimal DW design is described in section 5. Section 6 presents the implementation and experimentation. Finally, concluding remarks for this paper are drawn in section 7.

2. Data warehouse design

2.1 *Concepts of data warehouse*

The data warehouse (DW) can effectively aggregate massive amounts of data for analytic purposes, and provide valuable information for decision makers with a multi-dimensional view (Han and Kamber 2001). DW is not a single software or hardware product to provide enterprise information. It is, rather, a computing environment where users can find aggregated information. DW is a decision support oriented information technology. Inmon (1996) defined the DW as 'a subject oriented, integrated, nonvolatile, and time variant collection of data in support of management's decisions'. DW can also be viewed as a process for gathering, storing, managing, and analysing data (Gardner 1998). Further details of DW technology and business implementation can be found in Devlin (1997) and Paulraj (2001).

The three challenging issues for data warehousing are data acquisition, data storage and information delivery (Paulraj 2001). Data acquisition covers the entire process of extracting data from the data sources, moving all the extracted data to the staging area, and preparing the data for loading into the repository. Data storage covers the process of loading the data from the staging area into the repository. The information delivery spans a broad spectrum of different methods of making information available to users. Most of the information accessed in a DW is through online queries and interactive analysis sessions. OLAP (on-line analytical processing) is a query-based methodology that supports data analysis in a multi-dimensional environment. An OLAP engine logically structures multi-dimensional data in the form of a data cube in DW.

DW has become an essential technique for business intelligence. DW can easily integrate heterogeneous data derived from various systems, and manage data in compliance with user requirements. OLAP provides an interactive mechanism, which allows users to analyse the aggregated data in DW with a mixture of examinations by multi-dimensional viewing. Incorporated with OLAP, visualization and data mining, knowledge and patterns can be discovered from the massive data in DW. The multi-dimensional data model is adopted as a modelling technique for DW for decision support. Also, OLAP is designed for efficient data queries to assist decision-making. When transactional data are no longer of value to the operational environment they are removed from the database. If an enterprise does not possess a decision support facility, the data are archived and eventually destroyed. However, if a decision support system is built, the data can be transmitted to a type of interactive medium commonly referred to as a DW.

2.2 *DW design strategies*

A useful DW provides a flexible mechanism for users to view and analyse the massive amounts of data stored in the distributed heterogeneous systems. OLAP is constructed on the basis of multi-dimensional analysis. The system maintainability is another essential factor for a successful DW. The system maintainability challenges the DW designers due to the massive data growth and memory explosion once the DW comes into play. The memory space has a significant impact on efficiency, reliability and maintainability of DW. In order to efficiently display and visualise the aggregated information, data are generally categorized into a variety of multi-dimensional data views (cubes) (Gray *et al.* 1996). Data cube design and data materialization for data aggregation are also essential factors for effectiveness in DW.

Materialization in DW can generate a huge amount of data cubes. For example, a multi-dimensional view of 10 dimensions with three levels, as a maximum, consists of $(3+1)^{10}$ cubes. However, it is not realistic to materialize all cubes since the memory space is limited. There are therefore three cube selection strategies proposed as follows (Han and Kamber 2001):

1. *Full materialization:* this strategy materializes all of the cubes to efficiently respond to user queries. However, the DW maintenance cost is relatively high due to the huge amount of memory space required and frequent update once data refresh.
2. *Non-materialization:* this strategy performs data aggregation from the original data storage to respond user queries. The time to respond is prolonged, particularly in the case of data aggregation with multiple dimensions.
3. *Partial materialization:* this strategy can benefit from the trade-off between query response time and cube maintenance cost. Instead of full materialization, this strategy carefully selects a proper set of sub-cubes to perform materialization according to pre-selected criteria.

In DW, the appropriate partial materialization of views (sub-cubes) can balance the query response time and DW maintenance cost (Harinarayan *et al.* 1996). Harinarayan *et al.* developed a cost model to select a proper set of views for materialization. Known as a greedy search (GS) algorithm, this is adopted to optimize the selection of views within the query cost model. Ezeife (1997) and Gupta *et al*. (1997) additionally took the indexes of views into consideration for materialization. However, the DW maintenance cost was not considered in the cost models in Ezeife (1997) and Gupta *et al*. (1997).

Shukla *et al*. (1998) proposed a modified greedy algorithm to optimally select a set of views to materialize. They also demonstrated the superiority of their approach to that developed by Harinarayan *et al*. (1996). Furthermore, Shukla *et al*. (1998) addressed the problem of view selection for multi-cube data models. Smith *et al*. (1998) developed a dynamic method for view selection. The GS-based approaches were usually adopted in the previous studies (Harinarayan *et al*. 1996, Shukla *et al*. 1998). GS can generate cube selection designs with few computational requirements. However, the solutions may be far from the optima. More recently, Lin and Kuo (2000; 2004) developed a GA-based approach to data cube configuration. In their approach, a backward GS (BGS) repair mechanism is incorporated with GA to adjust for the infeasible solutions. Provided that the memory space is relatively tight, BGS + GA require much more iterations to correct the infeasible cube design. As a result, this paper proposes an approach, in which a forward GS (FGS) mechanism is incorporated with GA to resolve the cube selection problem in DW. The solutions obtained by FGS are taken as the initial solutions for GA to further polish the cube design.

3. Design of EMS DW

3.1 *User requirement analysis*

Nowadays, several equipment maintenance strategies, namely preventive maintenance, predictive maintenance and optimal maintenance, are applied to reduce maintenance cost and production loss, while increasing the equipment availability and utilization. An equipment management system (EMS) computerizes and automates the related maintenance tasks of equipment review, maintenance planning, work order, operator scheduling, spare part management, failure analysis and prediction. The basic functions of an EMS include quality control and management, vendor evaluation, maintenance reports, spare part inventory management, preventive maintenance, work order management and craftsmen management (Raouf *et al*. 1993). By implementing EMS, enterprises can benefit from productivity improvement, operation cost reduction, utilization increase, resource allocation optimization and maintenance quality enhancement.

The proposed EMS DW of this paper aims at providing an effective decision support system for equipment management. Here, an EMS DW is developed to support decision-making for equipment maintenance. First, the user requirements are analysed to determine the DW structure. Second, through on-line transactional processing (OLTP) database analysis, the data models are designed, and the required data are extracted from the database. Then thirdly, the data schema, fact tables and dimension tables are designed to build the EMS DW.

In EMS, equipment maintenance and spare part management are essential tasks in equipment management (MRO Software, Raouf *et al*. 1993). In equipment maintenance management, machine utilization, mean time to repair (MTTR), mean

Table 1. The necessary information of EMS.

Cubes	Indices	Information queried
Corrective maintenance	Machine, machine category	Machine utilization
	Machine, machine category, machine vendor, failure cause	MTTR
	Machine, machine category, machine supplier	MTBF
	Machine, machine category, location, operator, machine vendor, failure cause, failure cause	Number of maintenance in a time interval
	Machine, machine category, location, operator, machine vendor, failure cause, failure cause	Maintenance cost
Spare parts management	Spare part vendor, spare part, storage location	Average response time of rush parts
	Machine, machine category, spare part vendor, parts (items), storage location	Number of parts used
	Machine, machine category, spare part vendor, parts, storage location	Unused ratio of parts
	Machine, machine category, parts	Cost ratio of inventory to equipment

time between failures (MTBF), number of corrective maintenance in a time interval and maintenance cost can be adopted as measures for a data cube. In spare part management, response time of rush parts, number of parts (items) used, unused ratio of parts (cost of parts unused/cost of inventory), and cost ratio of inventory to equipment (cost of inventory/equipment investment) are all used as measures for analysis (Chen and Chiu 2003). To achieve the objective of near-zero downtime, EMS DW are incorporated with the DW of a shop floor operations management (SFOM) system to obtain the shop floor data such as yield, amount of inputs, cycle time, queue time, number of exceptions, exception time, and so on.

The necessary information of EMS is summarized in table 1. Maintenance engineers may invoke queries about machine utilization, MTTR, MTBF, average response time of equipment supplier, number of repairs to machines, and so forth. Additionally, spare part managers may make queries about such factors as average response time of vendors, number of parts used, unused ratio of parts, cost ratio of inventory to equipment, etc.

3.2 *Data warehouse schema*

Multi-dimensional queries provide various views into the data in DW, in which users can apply several data warehousing operations like consolidation, drill-down, drill-up, slicing and dicing to summarize and analyse data in a highly accurate

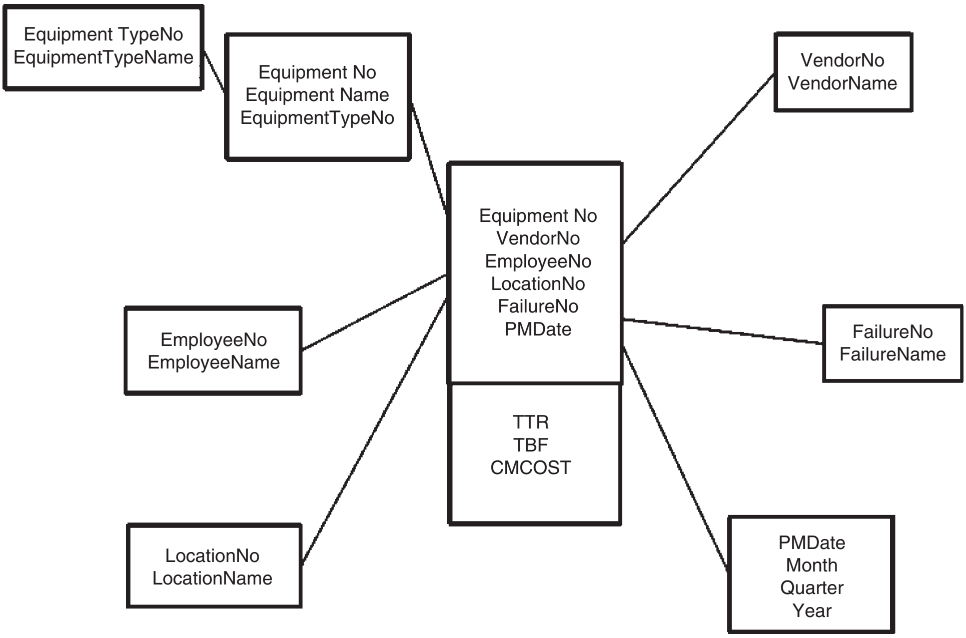

Figure 1. Data schema of EMM.

manner. The Star and Snowflake schemas are very concise and suitable for multi-dimensional queries. The multi-dimensional data cube is taken as the data repository.

Through the analysis of data schema, we then can select suitable dimensions and measures for EMS DW. With respect to the user requirements, 10 dimensions are included in the proposed EMS DW. They are equipment, equipment type, equipment location, equipment vendor, spare item, employee, failure, spare part vendor, part warehouse and time (preventive maintenance time and spare item order time). The measures of EMM include time to repair (TTR), time between failures (TBF) and corrective maintenance cost (CMCost). The measures of SPM are delivery lead time (delivery time), spare item cost (item cost), purchase order quantity (POQty), used quantity (UsedQty), overdue quantity (OverdueQty) and equipment investment (EqCost). The data schema, dimension tables and fact tables of EMM and SPM are respectively illustrated in figures 1 and 2.

4. The formulation of cube selection

For optimal selection of cubes (views), an optimization approach based on GA has been developed, and it is compared to existing alternatives for performance analysis. According to the analysis of user requirements for the EMS as mentioned above, the data cubes in the snowflake schema are designed for EMM and SPM, in which there exist six and five dimensions respectively. The full materialization is extremely expensive and impractical due to the memory space limitation and DW maintenance

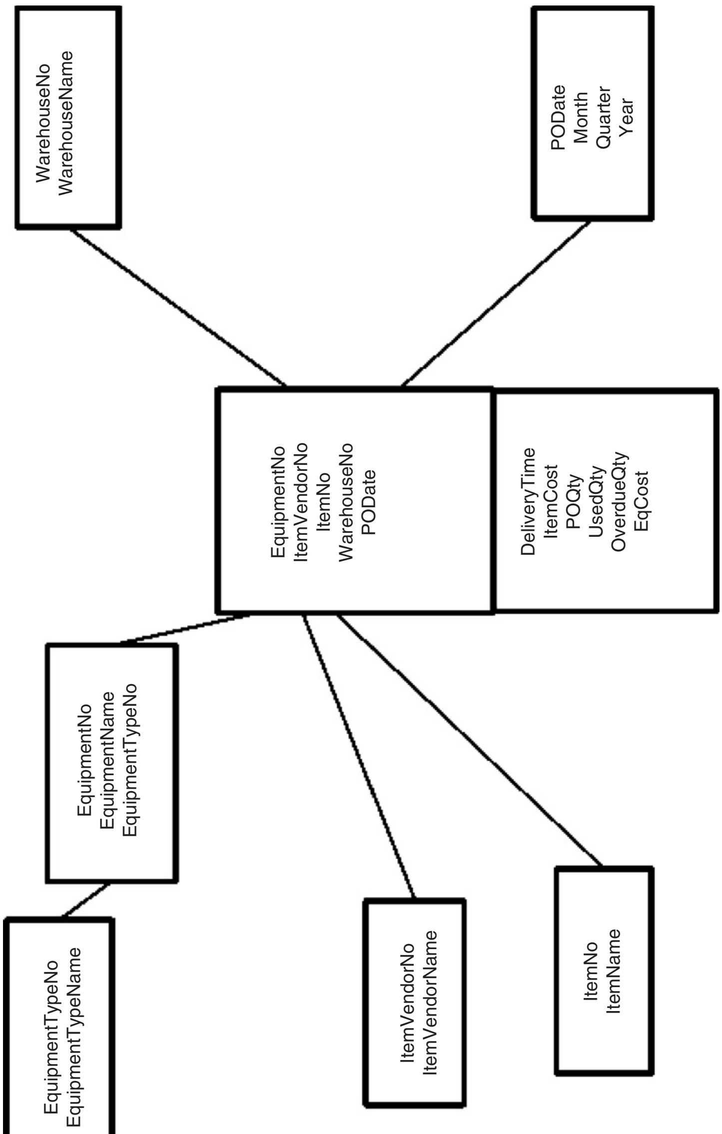

Figure 2. Data schema of SPM.

cost in EMS DW. Therefore, the partial materialization is suggested; however, it raises a problem of how to select a proper set of views for materialization.

4.1 *The cost model*

Before formulating the cost model, the nomenclature is firstly defined as follows:

Nomenclature

L	An OLAP cube lattice with n cubes.
$\mathbf{C}$	A set of cubes derived from a fact table, $\mathbf{C} = \{c_1, c_2, \ldots, c_n\}$.
$\mathbf{Q}$	A set of user queries, $\mathbf{Q} = \{q_1, q_2, \cdots, q_k\}$.
$\mathbf{F}$	A set of frequency values for each query, $\mathbf{F} = \{f_{q_1}, f_{q_2}, \ldots, f_{q_k}\}$.
$\mathbf{G}$	A set of update frequency values for each cube, $\mathbf{G} = \{g_{c_1}, g_{c_2}, \ldots, g_{c_n}\}$.
$\mathbf{M}$	A subset of $\mathbf{C}$ in current materialization, i.e. the set of materialized cubes.
$Cost_{TL}$	The total materialization cost, which includes the total query cost and total DW maintenance cost.
$Cost_{TQ}$	The total query cost.
$Cost_{TM}$	The total DW maintenance cost.
f_c	The invoking frequency of cube c, $f_c = \sum_{q \in Q,\, c=c_q} f_q$ in which the cube with just the same dimensions as query q be c_q.
g_u	The frequency of insertions to the fact table.
$E(c_i, \mathbf{M})$	The cost for evaluating cube c_i with respect to $\mathbf{M}$.
$U(c, \mathbf{M})$	The cost for updating cube c with respect to $\mathbf{M}$.
s_j	The required memory space of c_j.
S	The total memory space available.

The total cost, $Cost_{TL}$, consists of query cost, $Cost_{TQ}$, and DW maintenance cost, $Cost_{TM}$ (Lin and Kuo 2000, 2004). The query cost represents the cost caused by the aggregations functions in the OLAP analysis. The formulation of query cost defined in Harinarayan *et al.* (1996) is adopted herein. In Harinarayan *et al.* (1996) the query cost is equal to the number of non-null cells in the materialized cube used to answer the query. A linear relationship between the cube size and query evaluation time is investigated and confirmed by Harinarayan *et al.* (1996). The query cost takes the form (Harinarayan *et al.* 1996) as

$$Cost_{TQ} = \sum_{i=1}^{n} f_{ci} \times E(c_i, \mathbf{M}) \tag{1}$$

For the DW maintenance cost, the insertion of tuples into the fact table from which the cubes are derived is taken into consideration (Lin and Kuo 2000, 2004). The DW maintenance cost takes the form (Lin and Kuo 2000, 2004) as

$$Cost_{TM} = g_u \sum_{c \in \mathbf{M}} U(c, \mathbf{M}) \tag{2}$$

Hence, the total materialization cost can be expressed (Lin and Kuo, 2000, 2004) as

$$Cost_{TL} = \sum_{i=1}^{n} f_{ci} \times E(c_i, \mathbf{M}) + g_u \sum_{c \in \mathbf{M}} U(c, \mathbf{M}) \tag{3}$$

The cube selection problem attempts to minimize the total materialization cost under the space memory constraint. The constraint of memory space can be expressed as:

$$\sum_{s_i \in M} s_i \leq S \tag{4}$$

The dimensions and measures adopted in cubes are selected according to user requirement analysis. In practice each dimension should therefore be considered in one of the cubes, that is, at least one cube related to the dimension should be materialized. Hence, the number of dependent sub-cubes that are materialized should be at least one. This constraint takes the expression as

$$\left|(c, \mathbf{M})\right| \geq 1 \tag{5}$$

where $|(c, \mathbf{M})|$ represents the number of dependent sub-cubes that are materialized. In particular this constraint may be violated in the case of insufficient memory space.

5. The cube selection approach

The cube selection problem described in the previous section can be optimized to design a DW for effective equipment management. The cube selection is recognized as an NP-hard problem (Harinarayan *et al*. 1996, Lin and Kuo 2004). As a result, a GA-based cube selection approach as is proposed in this paper incorporates the greedy search (GS) to generate initial solutions for further evolution and improvement.

5.1 *Greedy search*

The greedy search (GS) approach was proposed by Harinarayan *et al*. (1996) for dealing with the cube selection problem. Since the maintenance cost was not considered in Harinarayan *et al*. (1996), GS iteratively select cubes until no cube can be included due to the memory space constraint. Harinarayan *et al*. (1996) then defined a benefit function to select the most appropriate cube to be included in each iteration. The benefit function $B(c, \mathbf{M})$ can be defined (Harinarayan *et al*. 1996) as

$$B(c, \mathbf{M}) = \frac{1}{S}\left\{\sum_{c_i \in \mathbf{M}} [E(c_i, \mathbf{M}) - E(c_i, \mathbf{M} \cup c)] + \sum_{c_i \in \mathbf{M}} [U(c_i, \mathbf{M}) - U(c_i, \mathbf{M} \cup c)]\right\} \tag{6}$$

where $E(c_i, \mathbf{M} \cup c)]\}$ and $U(c_i, \mathbf{M} \cup c)]\}$ respectively represent the query cost and maintenance cost when adding cube c for materialization. Hence $B(c,\mathbf{M})$ represents the saving of both query and maintenance costs per space if cube c is materialized. The forward GS (FGS) algorithm is presented as follows.

Forward greedy search algorithm

Step 0: $\mathbf{M}=\{c_s\}$, in which c_s is randomly selected from the set of sub-cubes $\mathbf{C}$.

Step 1: If $\sum_{s_i \in M} s_i \leq S$, then perform Steps 2 to 4. Otherwise proceed to Step 5.

Step 2: Calculate benefit function $B(c, \mathbf{M})$ for each candidate cube according to equation (6).

Step 3: Select the cube c with the maximal benefit.

Step 4: Add cube c for materialization, $\mathbf{M} \leftarrow \mathrm{M} \cup c$. Return to Step 1.

Step 5: If each dimension is considered in one of the sub-cubes ($|c, \mathbf{M}| \geq 1$), then go to Step 6. Otherwise return to Step 0 to restart the search.

Step 6: Terminate the search and return $\mathbf{M}$.

5.2 *Forward greedy search and genetic algorithm*

This subsection describes the hybrid approach based on FGS and genetic algorithm (GA), namely FGS + GA. The solutions obtained by FGS are taken as the initial solutions for GA to further polish the cube design. The algorithm of FGS+GA is presented as follows:

FGS + GA algorithm.
Step 0 (Initialization): Initialize the cube selection and algorithm-specific parameters.

Step 1 (Initial population generation): Generate an initial population $P(0)$, for GA by running FGS. Set $t=0$.

Step 2 (Termination test): If $t \leq$ maximum_generation, then perform Steps 3 to 7. Otherwise proceed to Step 8.

Step 3 (Fitness evaluation): Check the feasibility of each individual (cube selection), $\mathbf{M}_j \in P(t)$. If $\mathbf{M}_j$ is not a feasible solution then penalise $\mathbf{M}_j$. Compute the fitness value.

Step 4 (Selection): Select a pair of chromosomes as parent by roulette wheel selection. If $|O(t)| < population\_size$ (in which $O(t)$ is the offspring in generation t), then repeat Steps 5 and 6. Otherwise proceed to Step 7.

Step 5 (Crossover): Perform crossover mechanism.

Step 6 (Mutation): Perform mutation mechanism.

Step 7 (Replacement): Take the evolved offspring $O(t)$ as new population for next generation $P(t+1)$.

Step 8 (Termination): Terminate and return the optimal cube design.

The details of chromosome encoding, fitness evaluation and genetic operators are described as follows.

Encoding: The chromosome for cube design is encoded as a binary string. In a chromosome each gene (bit) represents the materialization condition of a sub-cube. The bit with '1' implies the sub-cube is selected for materialization. For example, a chromosome of eight genes, (10011010), is used to represent the materialization condition of eight sub-cubes. In this example, sub-cubes 1, 4, 5 and 7 are selected for materialization.

Fitness evaluation: The cube selection problem intends to minimize the total materialization cost. Therefore, the fitness value of cube design is the reciprocal of total cost, and it takes the form of

$$Fitness(\mathbf{M}_i) = \frac{1}{Cost_{TL}(\mathbf{M}_i)} \tag{7}$$

Selection: The roulette wheel selection (Goldberg 1989) is adopted in this paper. The selection probability of each chromosome is proportional to the fitness. The selection probability takes the form as

$$\Pr(c_i) = \frac{Fitness(\mathbf{M}_i)}{\sum Fitness(\mathbf{M}_i)} \tag{8}$$

Penalty function: The genetic operators may generate infeasible solutions, which violate the memory space and dependent sub-cube constraints. In this paper the penalty function has been adopted to force GA to locate feasible solutions. The penalty function takes the form of

$$Penalty(\mathbf{M}_i) = \begin{cases} LN \times \left(\sum_{j=1}^{n} s_j - S\right)^2, \text{if} \sum_{j=1}^{n} s_j > S; \\ LN, \text{if} \left|(c, \mathbf{M}_i)\right| = 0; \\ LN \times \left[\left(\sum_{j=1}^{n} s_j - S\right)^2 + 1\right], \text{if} \sum_{j=1}^{n} s_j > S \text{ and } \left|(c, \mathbf{M}_i)\right| = 0; \\ 0, \quad \text{otherwise.} \end{cases} \tag{9}$$

where LN is a relatively large positive number.

Crossover: The uniform crossover (Gen and Cheng 1997) is used herein. It randomly generates a mask which is a binary string with a similar length to that of the chromosome. The pair of parent chromosomes exchanges their genes if the corresponding values in the mask are '1'. Figure 3 schematically illustrates the uniform crossover by using an example.

Mutation: A mask is also randomly generated in the uniform mutation mechanism (Gen and Cheng 1997). The length of the mask is similar to that of the chromosome. The values in the mutation mask are between 0 and 1. The genes are mutated provided that the corresponding values in the mask are greater than the specified mutation rate. The flip-flop mechanism is adopted in the mutation operator. Figure 4 schematically illustrates the uniform crossover by using an example.

In the above-mentioned review papers (Aytug *et al.* 2003, Chaudhry and Luo 2005, Oduguwa *et al.* 2005), most production and operations management problems addressed by GAs are NP-hard and difficult to obtain the exact solutions. Due to the parallel adaptive nature, meta-heuristics (including GA) afford additional

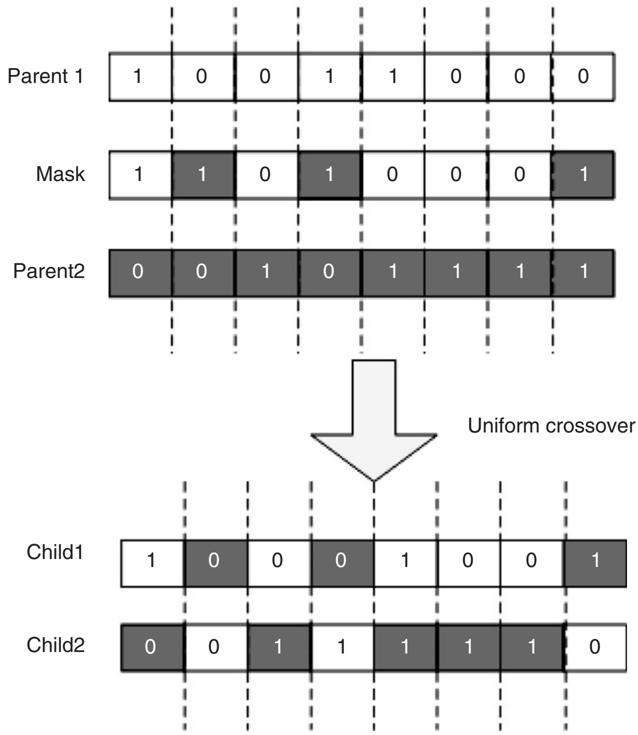

Figure 3. Illustration of uniform crossover.

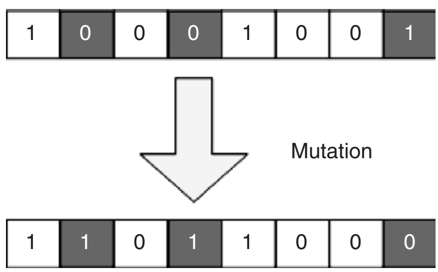

Figure 4. Illustration of uniform mutation.

optimization methods to conventional ones for resolving production problems (Oduguwa *et al*. 2005).

There are four main advantages when applying GAs to optimization problems (Gen and Cheng 1997). First, GAs can handle any kind of objective functions and any kind of constraints defined in discrete, continuous, or mixed search spaces. Second, the evolution operators make GAs very effective and reliable at finding nearly-global optima. Third, GAs can be easily incorporated with domain-dependent heuristics to enable the efficient implementation. Fourth, GAs have a high extensibility in algorithm design. Therefore, GAs have been successfully applied to resolve various difficult real-world problems (Goldberg 1989, Goldberg 1994, Gen and Cheng 1997).

Some researchers have tried to provide the theoretical foundation for the optimality of GAs. Greenhalgh and Marshall (2000) demonstrated that a GA can converge to the global optimum with any specific confidence level provided that a substantial amount of computational effort is afforded. They also theoretically built a bound for the number of required iterations to converge if some GA conditions are satisfied. Liang *et al*. (2001) performed the convergence analysis for an extended strings-based GA (ESGA) which is corresponding to the canonical GA (CGA). From their theoretical analysis, ESGA can reach a global optimum averagely with a finite computational requirement. Although the bound for number of required iterations and rate of convergence presented in Greenhalgh and Marshall (2000) and Liang *et al*. (2001) are met with some particular conditions, they support the primary underpinning of GAs in addressing the complex optimization problems.

The combinatorial optimization is a valuable area for the applications of evolutionary computation (EC) algorithms (Eiben and Schoenauer 2002). Furthermore, incorporating EC with traditional operational research approaches can obtain impressive improvement on some classical combinatorial optimization problems. Therefore, this paper presents an optimization approach (FGS + GA), which hybridizes GA and FGS, is used to optimally materialize the EMS DW. Provided that the memory space is relatively tight, BGS + GA developed by Lin and Kuo (2004) may require much more iterations to correct the infeasible cube design.

Nevertheless, some barriers of implementing meta-heuristics to the real-world optimization problems in industry were summarized by Oduguwa *et al*. (2005). From the review in Aytug *et al*. (2003), a conclusion was also drawn that very few previous studies had demonstrated the enthusiastic comparison results of GAs to alternative meta-heuristics in production and operations management problems. They referred to the no free lunch (NFL) theorems (Wolpert and Macready 1997) on the performance evaluation of meta-heuristics. According to NFL theorems, Greenhalgh and Marshall (2000) also indicated that GAs outperform other heuristics on some subsets of problems; others, however, can outperform GAs on problems out of the subsets. Aytug *et al*. (2003) pointed out some limitations in the effective evaluation of GAs. They additionally advised that the effectiveness of GAs should be substantially investigated according to the guidelines given in Barr *et al*. (1995) and Hooker (1995), who mainly suggested that the helpful evaluation of heuristics requires a set of well-defined experiments.

6. Implementation and experimentation

In order to test the performance of the proposed cube design approach, FGS + GA, a case of equipment management in a Taiwan semiconductor packaging manufacturer was applied for the purposes of this paper. There are 200 000 records related to the semiconductor manufacturing equipment, maintenance and spare part in the EMS database. In this case there are five plants with 34 categories of equipment such as implantation machine, wafer cutting machine, cleaning machine, laser printing machine, framing machine, etc. There are 21 suppliers to support the semiconductor equipment maintenance and spare part replenishment.

The EMS DW was developed on a Microsoft SQL 2000 Server and Analysis Service (Microsoft SQL 2000 OLAP). The dynamic visualization of OLAP is supported by PIVOT function in Microsoft Excel. Users can browse the OLAP results through Internet Explorer in the client tier.

6.1 *Equipment management data warehouse*

The EMS DW includes two schemas of equipment maintenance management (EMM) and spare part management (SPM) in the semiconductor packaging manufacturer. Figures 5 and 6 illustrate the schemas of EMM and SPM, respectively. In these two schemas, there are nine dimensions consisting of: equipment, failure, location, employee, equipment vendor, spare part storage, spare part item, spare part vendor and time (maintenance time and order time).

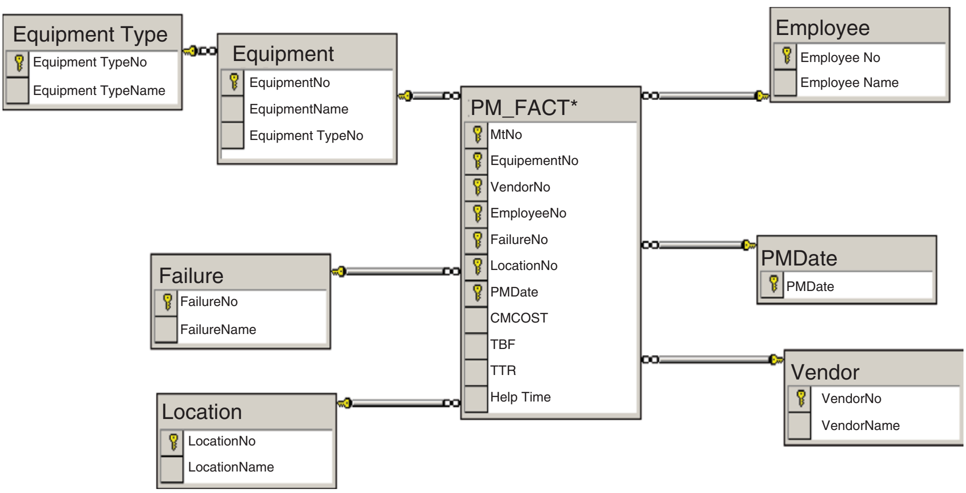

Figure 5. Cube schema of EEM.

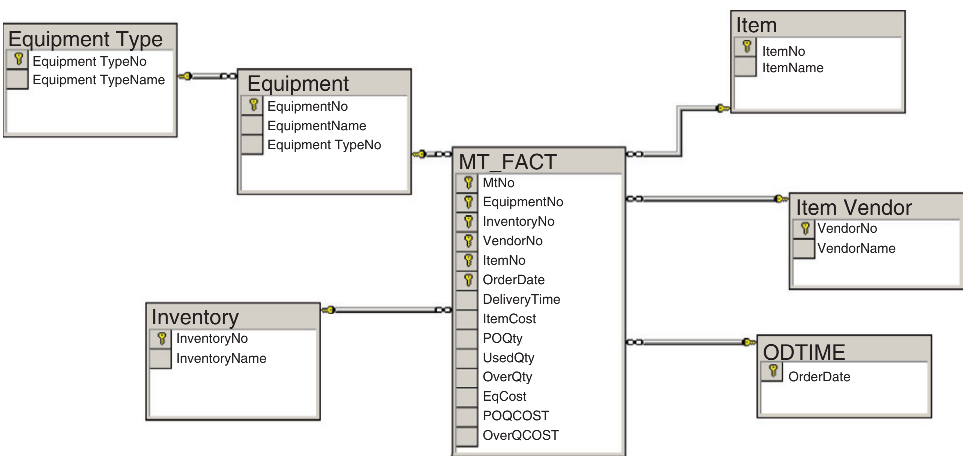

Figure 6. Cube schema of SPM.

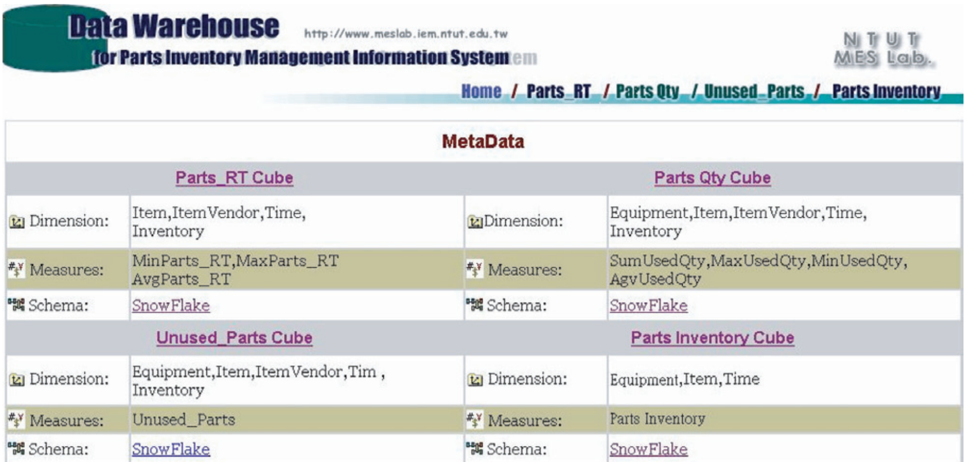

Figure 7. Illustration of EMS DW for semiconductor packaging manufacturing.

The measures in EMM include machine utilization, time to repair (TTR), time between failures (TBF), maintenance cost (CMCost), number of maintenance, and mean time to respond of vendor (MTRV). In SPM there exist four measures involving mean time to respond of spare part (MTRS), quantity of spare part (QSP), unused rate of spare part (URS) and storage rate of spare part (SRS). Figure 7 schematically illustrates the EMS DW for the semiconductor packaging manufacturing.

6.2 *Cube selection*

In EMM the cubes of equipment availability, MTTR, MTBF, equipment maintenance cost and maintenance frequency are built by using the schema presented in figure 5. In this subsection the cube of equipment maintenance cost (CMCost) in EMM was adopted to demonstrate the effectiveness of the proposed FGS + GA approach for optimal cube selection. Before the cube design procedure, the query frequency and required memory space of equipment maintenance cost were estimated for the 96 possible sub-cubes, and they are summarized in tables 2 and 3 respectively. Additionally, provided that no sub-cube is materialized for a specific query, it is necessary to aggregate the data from the original database. In such situations the query cost is much larger than those from materialized cubes. In tables 2 and 3, M, E, F, D, V and L respectively represent the dimensions equipment, employee, failure, maintenance time, vendor and location being selected in the GROUP BY statement in SQL. The symbol '-' implies the corresponding dimension is not used in the view.

In this paper we compared the cube selection results obtained by the proposed FGS + GA to those of FGS and BGS + GA developed by Lin and Kuo (2004). In BGS + GA, BGS was incorporated with GA to repair the infeasible solutions. Unlike FGS, BGS initially assumes all sub-cubes are materialized and then iteratively eliminates the sub-cube which has the least influence on the cube design.

Table 2. The required memory space for sub-cubes in equipment maintenance cost.

MEFDVL	200.000	M–DVL	185.438	M-FDVL	200.000	-E-F–	1.010
MEFDV-	195.952	M–DV-	72.144	M-FDS-	197.368	-E-DVL	184.813
MEFD-L	200.000	M–D-L	146.211	M-FD-L	200.000	-E-DV-	48.748
MEFD–	200.000	M–D–	26.316	M-FD–	187.485	-E-D-L	140.957
MEF-VL	38.956	M—VL	2.116	M-F-VL	28.499	-E–VL	0.750
MEF-V-	5.050	M—V-	0.075	M-F-V-	2.255	-E–V-	0.025
MEF–L	22.781	M—L	0.449	M-F–L	12.269	-E—L	0.150
MEF—	1.010	M—	0.015	M-F—	0.451	-E—	0.005
ME-DVL	196.149	MEFDVL	200.000	M–DVL	189.122	–FDVL	200.000
ME-DV-	139.124	MEFDV-	200.000	M–DV-	99.176	–FDV-	195.914
ME-D-L	185.303	MEFD-L	200.000	M–D-L	160.39	–FD-L	200.000
ME-D–	72.003	MEFD–	197.435	M–D–	44.869	–FD–	180.283
ME–VL	7.289	MEF-VL	41.560	M—VL	4.008	–F-VL	22.176
ME–V-	0.375	MEF-V-	10.644	M—V-	0.170	–F-V-	1.010
ME—L	2.123	MEF–L	29.095	M—L	1.008	–F–L	5.878
ME—	0.075	MEF—	2.255	M—	0.034	–F—	0.202
M-FDVL	200.000	ME-DVL	196.918	-EFDVL	200.000	—DVL	140.573
M-FDV-	195.914	ME-DV-	99.176	-EFD-L	200.000	—DV-	10.606
M-FD-L	200.000	ME-D-L	160.390	-EFD–	195.952	—D-L	57.035
M-FD–	180.283	ME-D–	44.869	-EF-V-	5.050	—D–	2.190
M-F-VL	22.176	ME–VL	4.008	-EF-SL	38.956	—VL	0.150
M-F-V-	1.010	ME–V-	0.170	-EF–L	22.781	—V-	0.005
M-F–L	5.878	ME—L	1.008	-EF—	10.604	—L	0.030
M-F—	0.202	ME—	0.170	-EDF-L	200.000	—	0.001

Table 3. The query frequency for sub-cubes in equipment maintenance cost.

MEFDVL	1000	M–DVL	454	M-FDVL	2000	-E-F–	222
MEFDV-	200	M–DV-	672	M-FDS-	1100	-E-DVL	222
MEFD-L	200	M–D-L	731	M-FD-L	555	-E-DV-	2000
MEFD–	200	M–D–	2000	M-FD–	5000	-E-D-L	222
MEF-VL	200	M—VL	232	M-F-VL	666	-E–VL	222
MEF-V-	200	M—V-	333	M-F-V-	345	-E–V-	222
MEF–L	200	M—L	404	M-F–L	231	-E—L	555
MEF—	200	M—	2000	M-F—	223	-E—	555
ME-DVL	200	MEFDVL	222	M–DVL	222	–FDVL	555
ME-DV-	200	MEFDV-	222	M–DV-	223	–FDV-	555
ME-D-L	200	MEFD-L	312	M–D-L	666	–FD-L	555
ME-D–	200	MEFD–	222	M–D–	2000	–FD–	876
ME–VL	200	MEF-VL	222	M—VL	999	–F-VL	665
ME–V-	200	MEF-V-	222	M—V-	777	–F-V-	888
ME—L	200	MEF–L	222	M—L	755	–F–L	222
ME—	20	MEF—	222	M—	2000	–F—	2000
M-FDVL	1000	ME-DVL	222	-EFDVL	222	—DVL	333
M-FDV-	2000	ME-DV-	222	-EFD-L	222	—DV-	444
M-FD-L	900	ME-D-L	222	-EFD–	222	—D-L	10
M-FD–	2000	ME-D–	222	-EF-V-	222	—D–	10
M-F-VL	1100	ME–VL	222	-EF-SL	222	—VL	150
M-F-V-	1200	ME–V-	222	-EF–L	222	—V-	666
M-F–L	140	ME—L	222	-EF—	222	—L	254
M-F—	202	ME—	222	-EDF-L	222	——	1

For the memory space constraint, a set of nine different space limits was used to test the effectiveness of FGS, BGS + GA and FGS + GA. The space limitations were set at 10%–90% of the total size of all sub-cubes. For FGS + GA and BGS + GA, several GA-specific parameters were defined by using some pilot runs with various settings. After the pilot runs these parameters were set to: *maximum_generation* = 150, *population_size* = 50, *crossover_rate* = 1.0 and *mutation_rate* = 0.2. Due to the stochastic nature of GA, 30 runs were performed to obtain the statistics of cube design.

The computational results with various space limits by FGS, BGS + GA and FGS + GA are summarized in tables 4 to 6, respectively. In tables 5 and 6, U_{mean}, U_{max}, U_{min} and U_{std} represent average cost, maximum cost, minimum cost and standard deviation of cost in cube design. As can be seen from these tables, the results indicate that it is more difficult to find a suitable cube design in cases which have tight space limits. Additionally, tables 7 and 8 respectively present the ratios of total cost of FGS to FGS + GA and BGS + GA to FGS + GA, and the ratios of CPU time of FGS to FGS + GA and BGS + GA to FGS + GA.

Figure 8 illustrates the comparison of total costs by FGS, BGS+GA and FGS+GA. The computational requirements of BGS+GA and FGS+GA are shown in figure 9. The CPU time of FGS is relatively short (less than 5 seconds). In the cases of FGS+GA and BGS+GA, the average cost and average CPU time are presented in figures 8 and 9.

Table 4. Computational results of FGS with various space limits.

	90%	80%	70%	60%	50%
Total cost	4964.65	4964.65	4964.65	4980.97	5488.12
	40%	30%	20%	10%	
Total cost	6674.15	8995.68	13138.75	20372.88	

*The CPU time of FGS is less than 5 seconds.

Table 5. Computational results of BGS + GA with various space limits.

	90%	80%	70%	60%	50%
U_{mean}	4928.86	4928.55	4928.86	4929.32	4935.12
U_{max}	4931.16	4929.40	4930.67	4930.91	4940.81
U_{min}	4928.48	4928.48	4928.48	4928.48	4931.95
U_{std}	0.67	0.19	0.55	0.87	2.17
CPU time (sec.)	39	46	65	113	191
	40%	30%	20%	10%	
U_{mean}	4949.49	4979.32	5134.24	6484.78	
U_{max}	4964.19	4994.24	5217.65	6837.31	
U_{min}	4942.73	4955.30	5059.75	6108.66	
U_{std}	4.89	10.56	44.70	280.60	
CPU time (sec.)	328	543	909	1289	

From tables 4 to 8 and figures 8 and 9, although FGS search required a shorter CPU time to locate a cube design solution, the solution quality was much inferior. In the cases with smaller spaces, particularly, solutions generated by FGS were far from those by BGS + GA and FGS + GA. From the computational results, FGS + GA and BGS + GA outperform FGS in terms of solution quality. Except for the case with a 10% space limit, FGS + GA and BGS + GA perform evenly in terms of solution quality. Where there are very tight space limits, BGS + GA generate better solutions than FGS + GA. However, BGS + GA generally require much more computational requirements than FGS + GA (see table 8). As mentioned above,

Table 6. Computational results of FGS+GA with various space limits.

	90%	80%	70%	60%	50%
U_{mean}	4928.48	4928.48	4928.48	4930.10	4934.50
U_{max}	4928.48	4928.48	4928.48	4933.76	4938.90
U_{min}	4928.48	4928.48	4928.48	4928.48	4931.36
U_{std}	0.00	0.00	0.00	2.06	2.35
CPU time (sec.)	36	36	37	36	38
	40%	30%	20%	10%	
U_{mean}	4953.27	4990.11	5130.48	8936.43	
U_{max}	4966.17	5020.74	5230.52	14600.21	
U_{min}	4944.76	4967.23	5068.52	5468.75	
U_{std}	6.93	15.39	42.37	2549.07	
CPU time (sec.)	38	38	39	37	

Table 7. Total cost ratios of FGS to FGS + GA and BGS + GA to FGS + GA.

	90%	80%	70%	60%	50%
FGS/FGS + GA	1.00734	1.00734	1.00734	1.01032	1.11219
BGS + GA/FGS + GA	1.00008	1.00001	1.00008	0.99984	1.00013
	40%	30%	20%	10%	
FGS/FGS + GA	1.34742	1.80270	2.56092	2.27976	
BGS + GA/FGS + GA	0.99924	0.99784	1.00073	0.72566	

Table 8. CPU time ratios of FGS to FGS + GA and BGS + GA to FGS + GA.

	90%	80%	70%	60%	50%
FGS/FGS + GA	0.13889	0.13889	0.13514	0.13889	0.13158
BGS + GA/FGS + GA	1.08333	1.27778	1.75676	3.13889	5.02632
	40%	30%	20%	10%	
FGS/FGS + GA	0.13158	0.13158	0.12821	0.13514	
BGS + GA/FGS + GA	8.63158	14.28947	23.30769	34.83784	

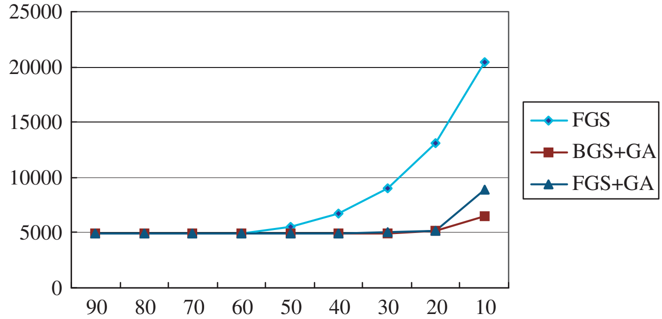

Figure 8. Comparison of three approaches in total cost.

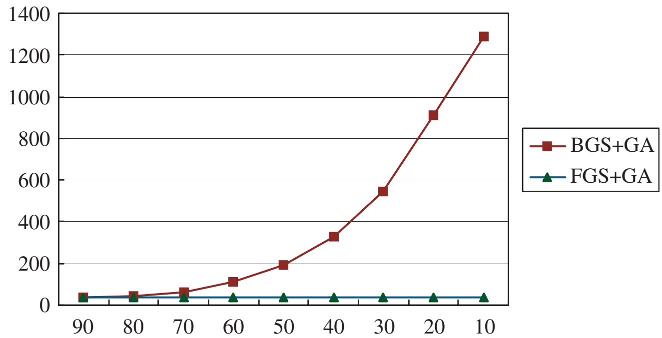

Figure 9. Comparison of BGS + GA and FGS + GA in CPU time.

MI----	MI---L	MI--V-	MI--VL	MI-D--	MI-D-L	MI-DV-	MI-DVL
M-F---	M-F--L	M-F-V-	M-F-VL	M-FD--	M-FD-L	M-FDV-	M-FDVL
M-----	M----L	M---V-	M---VL	M--D--	M--D-L	M--DV-	M--DVL
EIF---	EIF--L	EIF-V-	EIF-VL	EIFD--	EIFD-L	EIFDV-	EIFDVL
EI----	EI---L	EI--V-	EI--VL	EI-D--	EI-D-L	EI-DV-	EI-DVL
E-F---	E-F--L	E-F-V-	E-F-VL	E-FD--	E-FD-L	E-FDS-	E-FDVL
E-----	E----L	E---V-	E---VL	E--D--	E--D-L	E--DV-	E--DVL
-IFDV-	-ID---	-IF--L	-IF-SL	-IF-V-	-IFD--	-IFD-L	-IFDVL
-I----	-I---L	-I--V-	-I--VL	-I-D-L	-I-DV-	-I-DVL	-I-F--
--F---	--F--L	--F-V-	--F-VL	--FD--	--FD-L	--FDV-	--FDVL
------	-----L	----V-	----VL	---D--	---D-L	---DV-	---DVL

Figure 10. The optimal cube design for equipment maintenance cost.

BGS + GA may require much more iterations to correct the infeasible cube design provided that the memory space is relatively tight.

Taking the cube of equipment maintenance cost (CMCost) with a 30% space limit as an example, the optimal cube design is summarized in figure 10. In this figure the sub-cubes highlighted in grey are materialized. The other cubes of EMS are also designed by using the proposed FGS+GA. The query response times are then experimented upon by using 15 queries. As seen from figure 11, the response times of full materialization and partial materialization are close in these 15 queries. Table 9 lists and compares the memory spaces of five cubes by full

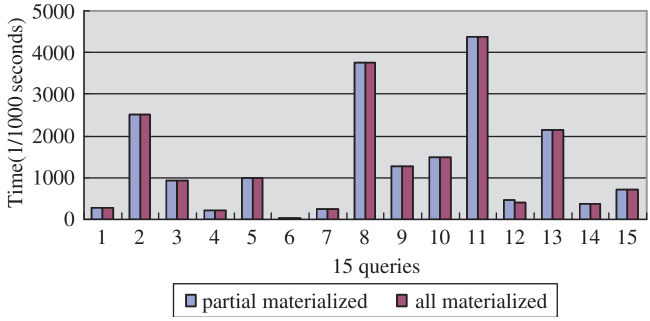

Figure 11. The response times of full materialization and partial materialization.

Table 9. The memory space of five cubes by full materialization and partial materialization.

	Full materialized	Partial materialized	Partial/Full
Cost cube	7685.073	2284.714	29.7%
Frequency cube	7685.073	2280.657	29.6%
MTTR cube	1397.988	356.241	25.4%
MTBF cube	255.601	156.424	61.1%
Availability cube	73.425	47.109	64.1%

materialization and partial materialization. The query response times of partial materialization and full materialization are similar. However, the required memory spaces of partial materialization are much less than that of full materialization. The proposed FGS+GA approach is also demonstrated to be a practical design approach for EMS DW to help equipment engineers make maintenance decisions with various equipment related dimensions for effective equipment management in semiconductor packaging manufacturing.

7. Conclusions

Effective equipment management is one of the important tasks for modern manufacturing companies in order to keep their competitive edge. The rapid advancements in e-manufacturing accelerate the enormous growth of equipment engineering data, which can be integrated and aggregated by data warehousing for supporting effective e-maintenance and e-diagnostics. In this paper a data warehouse for equipment management system, or EMS DW, was developed for a semiconductor packaging manufacturer. EMS DW could easily integrate heterogeneous data derived from various shop floor systems and manage data complied with OLAP for decision-making purposes. EMS DW allowed equipment engineers to analyse the aggregated equipment data with multi-dimensional views. This paper not only develops a GA-based approach, namely FGS + GA, to optimally select cubes (views), but also shows an implementation for designing the EMS DW for the semiconductor packaging manufacturing. According to the experimental results, the partial materialization suggested by the proposed FGS + GA approach can decrease by 70% the memory space of full materialization, while the query response times are approximately unchanged. The further performance evaluation of the proposed heuristic with respect to the guidelines recommended in Barr *et al*. (1995) and Hooker (1995) is one part for future work.

Acknowledgement

This work is partially supported by National Science Council, Taiwan, ROC under grant NSC 91-2213-E-027-020.

References

Aytug, H., Khouja, M. and Vergara, F.E., Use of genetic algorithms to solve production and operations management problems: a review. *Int. J. Prod. Res*., 2003, **41**, 3955–4009.

Barr, R.S., Golden, B.L. and Kelly, J.P., Resende, M.G.C. and Stewart, W.R., Designing and reporting on computational experiments with heuristic methods. *J. Heuristics*, 1995, **1**(1), 9–32.

Chaudhry, S.S. and Luo, W., Application of genetic algorithms in production and operations management: a review. *Int. J. Prod. Res*., 2005, **43**(19), 4083–4101.

Chen, K.-Y. and Chiu, T.-S., Deploying a data warehouse for equipment-related management information system, in *Proceedings of the 8th Annual International Conference on Industrial Engineering, Theory, Applications and Practice*, Las Vegas, USA, 2003.

Chen, M.-C., Roundness inspection strategies for machine visions using nonlinear programs and genetic algorithms. *Int. J. Prod. Res*., 2000, **38**(13), 2967–2988.

Chiang, T.-L. and Su, C.-T., Optimisation of TQFP molding process using neuro-fuzzy-GA approach. *Euro. J. Oper. Res*., 2003, **147**(1), 156–164.

Devlin, B., *Data Warehouse from Architecture to Implementation*, 1997 (Addison-Wesley Reading: Boston: MA).

Eiben, A.E. and Schoenauer, M., Evolutionary computing. *Inform. Proc. Letters*, 2002, **82**, 1–6.

Ezeife, C.I., A uniform approach for selecting views and indexes in a data warehouse, in *International Database Engineering and Applications Symposium*, 1997, pp. 151–160.

Gardner, S.R., Building the data warehouse. *Comm. ACM*, 1998, **41**(9), 52–60.

Gen, M. and Cheng, R., *Genetic Algorithms and Engineering Design*, 1997 (John Wiley & Sons: New York: NY).

Goldberg, D., *Genetic Algorithms in Search, Optimisation and Machine Learning*, 1989 (Addison-Wesley Reading: Boston: MA).

Goldberg, D.E., Comm. ACM, 1994, **37**(3), 113–119.

Gray, J., Bosworth, A., Layman, A. and Pirahesh, H., Data cube: a relational aggregation operator generalising group-by, cross-tabs and subtotals, in *Proceedings of the International Conference on Data Engineering*, 1996, pp. 152–159.

Greenhalgh, D. and Marshall, S., Convergence criteria for genetic algorithms. *SIAM J. Computing*, 2000, **30**(1), 269–282.

Gupta, H., Harinarayan, V., Rajaraman, A. and Ullman, J.D., Index selection for OLAP, in *Proceedings of ICDE*, 1997, pp. 208–219.

Han, J. and Kamber, M., *Data Mining: Concepts and Techniques*, 2001 (Morgan Kaufmann Publishers: San Francisco: CA).

Harinarayan, V., Rajaraman, A., & Ullman, J.D., Implementing data cubes efficiently, in *ACM SIGMOD'96*, 1996, pp. 205–216.

Holland, J.H., *Adaptation in Natural and Artificial System*, 1975 (The University of Michigan Press: Ann Arbor: MI).

Hooker, J.N., Testing heuristics: we have it all wrong. *J. Heuristics*, 1995, **1**(1), 33–42.

Hsu, C.-M., Chen, K.-Y. and Chen, M.-C., Batching orders in warehouses by minimising travel distance with genetic algorithms. *Computers in Industry*, 2005, **56**(2), 169–178.

Hung, M.-H., Cheng, F.-T. and Yeh, S.-C., Development of a Web services based e-diagnostics framework for semiconductor manufacturing industry. *IEEE Trans. Semicon. Manuf.*, 2005, **18**(1), 122–135.

Inmon, W.H., *Building the Data Warehouse*, 1993 (John Wiley & Sons: New York, NY).

Intelligent Maintenance Systems. Web site of the Center for Intelligent Maintenance Systems (IMS). Available online at: http://www.imscenter.net/ (accessed 18 May 2005).

Jeong, J., Leon, V.J. and Villalobos, J.R., Integrated decision-support system for diagnosis, maintenance planning, and scheduling of manufacturing systems. *International Journal of Production Research*, 2006, **45**, 267–285.

Liang, Y., Zhou, C., Wang, Z., Lee, H. P. and Lim, S. P., An equivalent genetic algorithm based on extended strings and its convergence analysis. *Inform. Sci.*, 2001, **138**, 119–135.

Lin, W.Y. and Kuo, I.C., OLAP data cubes configuration with genetic algorithms, in *Proceedings of the IEEEE SMC'00*, 2000, pp. 1984–1989.

Lin, W.Y. and Kuo, I.C., A genetic selection algorithm for OLAP data cubes. *Knowledge and Information Systems*, 2004, **6**(1), 83–102.

MRO Software, Maximo Software Introduction, 2002. Web site of the MRO software. Available online at: http://www.mro.com (accessed 20 May 2005).

Oduguwa, V., Tiwari, A. and Roy, R., Evolutionary computing in manufacturing industry: an overview of recent applications. *Appl. Soft Computing*, 2005, **5**, 281–299.

Paulraj, P., *Data Warehousing Fundamentals*, 2001 (John Wiley & Sons: New York: NY).

Raman, R., E-diagnostics in IC manufacturing: A case study. *Semicon. Int.*, 2001. Available online at: http://www.e-insite.net/semiconductor/ (accessed 18 May 2005).

Raouf, A., Ali, Z. and Duffuaa, S.O., Evaluating a computerised maintenance management system. *Int. J. Oper. & Prod. Manage.*, 1993, **13**(3), 38–48.

Shukla, A., Deshpande, P.M. and Naughton, J.F., Materialised view selection for multi-dimensional datasets, in *Proceedings of the 24th VLDB Conference*, 1998, pp. 488–499.

Smith, J.R., Castelli, V., Jhingran, A and Li, C.-S., Dynamic assembly of views in data cubes, in. *ACM PODS'98*, 1998 (Seattle: Washington, USA).

Su, C.-T. and Chiang, T.-L., Optimal design for a ball grid array wire bonding process using a neuro-genetic approach. *IEEE Trans. Electron. Pack. Manuf.*, 2002, **25**(1), 13–18.

Su, C.-T. and Shiue, Y.-R., An intelligent scheduling controller for shop floor control systems: a hybrid genetic algorithm/decision tree learning approach. *Int. J. Prod. Res.*, 2003, **41**(12), 2619–2641.

Wolpert, D.H. and Macready, W.G., No free lunch theorems for optimisation. *IEEE Trans. Evol. Compu.*, 1997, **1**, 67–82.

Wu, X., Chu, C.-H., Wang, Y. and Yan, W., Concurrent design of cellular manufacturing systems: a genetic algorithm approach. *Int. J. Prod. Res.*, 2006, **44**(6), 1217–1241.