Proceedings of the 10th
World Congress on Intelligent Control and Automation
July 6-8, 2012, Beijing, China

A Memetic PSO based KNN Regression Method for Cycle Time Prediction in a Wafer Fab

JiaCheng Ni, Fei Qiao, Li Li, Qi Di Wu
School of Electronics and Information Engineering
Tongji University
Shanghai 201804, China
nijiacheng@hotmail.com

***Abstract* – In this paper, cycle time prediction of wafer lots is studied. A memetic algorithm called GSMPSO by combining the PSO with a Gaussian mutation operator and a Simulated Annealing (SA)-based local search operator is developed to weight the features for K Nearest Neighbors (KNN) regression. The GSMPSO-KNN regression method is used to predict the cycle time of wafer lots. The experiment result demonstrates that a more accurate result can be obtained by the proposed method compared with some other prediction methods. The critical factors affecting the cycle time of wafer lots can also be extracted by the proposed method.**

Index Terms* – *cycle time prediction, PSO, memetic algorithm, Gaussian mutation, local search

I. INTRODUCTION

Cycle time prediction of wafer lots is a critical task to a semiconductor fab. Traditionally, the cycle time of a wafer lot is estimated based on the finished wafer lots with the same process routine. For example, the total work content (TWK) method [1-3]. The major disadvantage of the traditional method is the lack of prediction accuracy. The cycle time of a wafer lot also is affected by some other factors such as fab status. Therefore, this problem has attracted many researchers from academia and industry to develop accurate prediction method to meet the practical requirements.

In the literature, there are 3 techniques commonly used. The linear regression method [4] is a direct improvement of the traditional method. It has been widely used in practice because of its quickness and easiness. But it stills lacks accuracy. The most accurate method is the production simulation. A discrete-event simulation model is built for the fab and the cycle time of wafer lots is estimated by the simulation. But it is time consuming for the large scale fab. The artificial neural network (ANN) [5] method can be considered as a compromise between the accuracy and the time cost. The training examples are generated by the simulation or drawn from the real fab. The ANN is trained by the training algorithm and training examples. The time consuming work can be finished offline and the online prediction is quick. To relieve overfit, the training examples can be preprocessed by data clustering. Each cluster is used as a training set to train a neural network and a satisfactory accuracy can be achived by the ensemble of the ANNs trained by different data clusters [6]. Although the ANN can improve the accuracy compared with the linear regression and the traditional method, the improvement is obtained at the cost of comprehensibility because of the poor interpretability of ANN.

Particle Swarm Optimization (PSO) [7] is a relatively new evolutionary algorithm which reflects the swarm intelligence. PSO has become an active field of optimization techiques but it is vulnerable to the local optima when solving the complex multi-modal problems. To enhance PSO's ability of local search and escaping from the local optima, a memetic algorithm called GSMPSO is developed by combining the PSO with the Gaussian mutation operator and SA-based local search in our previous work [8]. GSMPSO is used to weight the features for KNN regression. The GSMPSO-KNN regression method is used to predict the cycle time of wafer lots. The result shows that the proposed method can reduce the prediction errors compared with the KNN and ANN based methods. The critical factors affecting the cycle times of the lots can also be extracted by the proposed method. This is useful for the fab staffs to improve the performance of planning and scheduling.

The rest of the paper is organized as follows: In Section II, we introduce a production simulation model for a real fab, the training examples are collected by simulation. In Section III, we investigated the Gaussian mutation and the SA-based local search, GSMPSO is proposed. In section IV, GSMPSO is used to weight the features for KNN regression. The GSMPSO-KNN regression is used to predict cycle time of wafer lots and compared with other methods in section V. Finally, we give the conclusion and future research topics in section VI.

II. PROBLEM DESCRIPTION

A. Simulation Model

In our previous work, the planning and scheduling problem of a real-world wafer fab in a wafer manufacutring factory located in Shanghai was investigated. There are about 200 machines to process the front-end process routines of 6-inch semiconductor wafers. Several product types of wafers are produced synchronously in the fab. For each product type, 300-400 steps are involved.

We build a discrete-event simulation model for the fab by eM-Plant 7.5 to aid the planning and scheduling staffs. The production model simulates the planning and scheduling activity of the fab and records the related performances. Some release control strategies and dipatching rules are implemented to optimize the performance of planning and scheduling. In

978-1-4673-1398-8/12/$31.00 ©2012 IEEE

this work, the simulation model is used to generate the training example set and testing data set.

In the simulation, the CONWIP (the WIP is a constant in the fab) release control method is used to keep a stable production environment. The rules of thumb used by the planning and scheduling staffs are used in the simulation to reflect the real scheduling of the fab. After a warm up period of simulation, 800 records are collected and divided into a training example set of 600 records and a testing data set of 200 records.

B. Specification of the Training Examples

In the literature, the factors affecting the cycle time of wafer lots have been widely discussed and investigated [1-6]. These factors can be classified into 3 categories: (1) the planning and scheduling strategies; (2) the attributes of the wafer lot; (3) the features describing the status of the fab when the wafer lot is released to the fab. As we assumed in last section that the release control strategy and dispatching rules are constant during the simulation to generate the training examples and testing data, the features in (1) are not considered in this work. The following 11 factors are selected to describe the specification of the example e_i for lot i and we suppose that the release time of lot i is t.

- WF_t: the number of work-in-process (WIP) in the fab at time t. In this work because the CONWIP release control strategy is used and the WIP is a constant of 312 which is a moderate loading level, WF_t is not considered in the experiment
- WBN_t: the number of the wafer lots in the bottleneck work area at time t.
- WBT_t: the number of the wafer lots in the batch process work area at time t.
- QF_t: The total queue length in the whole fab at time t.
- QBN_t: The total queue length before bottleneck work area at time t.
- QBT_t: The total queue length before batch process work area at time t.
- NRB_i: The number of reentrance of the bottleneck work area required for lot i's process routines.
- NOP_i: The number of the operations required for lot i's process routine.
- SK_i: The slack time of lot i.
- TP_i: The total processing time of lot i required for lot i's process routine.
- NSL_i: The number of lots with the same product type of lot i.

The value of e_i is the actual cycle time of lot i obtained by simulation.

III. MEMETIC PSO

In this section, the GSMPSO is discussed in detail. A memetic algorithm is an evolutionary algorithm that includes one or more local search operators to improve the individuals within its evolution cycles [9]. In order to design an effective and efficient MA, a scheme that the application of different operators depends on the particles' status is adopted. To illustrate this scheme, we classify the particles according to their quality and history. The Gaussian mutation operator, the SA-based local search operator and GSMPSO are described.

A. Particle Swarm Optimization

In PSO, the swarm consists of particles. Each particle represents a possible solution and its position and velocity are represented as $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$ and $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$. The index i represents the ith particle, D is the dimensionality. The movement of the ith particle is determined by v_i. x_i and v_i are adjusted by (1) and (2)

$$v_{id} = \chi \times \left(v_{id} + c_1 \times rand() \times (pbest_{id} - x_{id}) + c_2 \times rand() \times (gbest_{id} - x_{id}) \right) \quad (1)$$

$$x_{id} = x_{id} + v_{id} \quad (2)$$

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, where\ \varphi = c_1 + c_2 > 4 \quad (3)$$

The index d denotes the dth dimension. is called the constriction factor defined as (3) which is used to control the magnitude of the velocity. c_1 and c_2 are called the acceleration coefficients. $rand()$ can generate a random real number between 0 and 1 according to the uniform distribution. The position is evaluated by the objective function f. In this paper, we consider the minimization of f. $pbest_i$ is the best position obtained by the ith particle, and this part known as the cognition component. $gbest_i$ is the best position obtained by the ith particle's neighbors, and this part know as social component. (1) and (2) are used iteratively to update each particle's position and velocity to search the global optima .

B. Classification of the Particles

In GSMPSO, the mutation operator and the local search operator are incorporated into evolution cycles. In our assumption, the PSO responsible for the global search of the search space, the Gaussian mutation operator is applied to the stagnant particle to escape from the local optima and the SA based local search operator performs a fine-grained search around the promising regions detected by the population. The variables are introduced to each particle, i.e., $no\_improve_i$ reflecting the degree of stagnation of i and $improve_i$ detecting the promising regions.

- For each particle i, a variable $no\_improve_i$ is used to record the number of generations that the $pbest_i$ has not been improved, i.e., $pbest_i$ has not been changed for $no\_improve_i$ generations. If $no\_improve_i \geq 20$, i is a stagnant particle.

- For the ith particle, a variable $improve_i$ is used to record the number of generations that the $pbest_i$ has been improved continuously, i.e., $pbest_i$ has been changed continuously for $improve_i$ generations.
- A particle i with the best $pbest_i$ in the swarm is a promising particle if $improve_i \geqslant 3$.

C. Gaussian Mutation Operator

Based on the classification of the particles, some operators will be applied to the particles to improve the performance of the PSO. To prevent the premature convergence, a Gaussian mutation operator is designed and applied to the stagnant particles which tend to be trapped into the local optima. The Gaussian mutation operator used in this work is defined as (4) where $N(0,1)$ can generate a random real number subject to standard Gaussian distribution. The stagnant particle i is reinitialized by a perturbation of the $pbest_i$, and the variable $no\_improve_i$ and the velocity vector of i are reset to 0. In particular, each dimension of i is perturbed by (4), and the stagnant particle will restart to exploit the solution space from a new position which is a coarse-grained perturbation of $pbest_i$. The Gaussian mutation adopted in this paper can be considered as an adaption of the mutation operator defined in [10] which has been proved to be effective in [11].

$$\begin{aligned} pbest_{id} &= pbest_{id}(1+N(0,1)) \\ x_{id} &= pbest_{id} \\ v_{id} &= 0 \\ no\_improve_{id} &= 0 \end{aligned} \tag{4}$$

D. SA-based Local Search Operator

Kenndy developed a social science explanation for PSO from two aspects: One is the scheme of information change among the particles and the other is the PSO models [12] [13]. In this paper, the *lbest* topology in which each particle learns from the best position found by its left neighbour or right neighbour is adopted because the diversity can be preserved within its evolution cycles. But the problem of lacking the ability of local refinement remains obvious in *lbest* and this problem can lead to a miss of the local optima. To tackle this tough problem, a novel local search operator in which the cognition-only model [13] is incorporated into the SA framework is developed to enhance the exploitation ability of the *lbest*. The SA-based local search operator is only applied to the promising particles and the pseudocode of this SA-based local search operator is described in Fig.1.

$$v_{id} = w \times v_{id} + c_1 \times rand() \times (pbest_{id} - x_{id}) \tag{5}$$

In Fig.1, T is the temperature variable and T_0 is the initial temperature. *ls_length* is a constant which represents the number of iterations required by *SA_local_search*. $pbest_i'$ can be obtained by introducing a Cauchy perturbation to the rth dimension of $pbest_i$ according to (5) in which $[A_r, B_r]$ is the range of *the* rth parameter, and u is generated randomly subject to the uniform distribution between 0 and 1. Only one dimension of the $pbest_i$ is perturbed for the purpose of "fine-grained" local search around the promising regions. The $pbest_i$ is updated by a greedy search but the new position x_i' which is generated by the cognition-only model is accepted subject to the Metropolis rule [14]. A thorough local search around the promising region can be performed. Thus, the ability of local refinement of PSO can be enhanced by the *SA_local_search*.

$$\begin{aligned} pbest_{ir}' &= pbest_{ir} \\ &\quad + T\,\mathrm{sgn}(u-0.5)[(1+1/T)^{|2u-1|}-1](B_r - A_r) \\ pbest_{ir}' &\in [A_r, B_r] \end{aligned} \tag{6}$$

```
procedure SA_local_search(particle i)
begin
  initilize T=T0
  for j=0 to ls_length
      choose an integer r randomly from [1,D]
      pbesti' is generated by perturbing the rth dimension
      value of pbesti according to the Cauchy distribution
      vi is updated by vid = w*vid+c1*rand()*(pbestid' -xid)
      xi' is generated by xid' = xid+vid
      Δf=f(xi' )-f(xi)
      if(Δf<0)
        xi is updated by xi'
      else if(rand()<exp(-Δf/T))
        xi is updated by xi'
      endif
      if(f(pbesti' )<f(pbesti))
        pbesti is updated by pbesti'
      endif
      if(f(xi' )<f(pbesti))
        pbesti is updated by xi'
      endif
      T= T0/lg(1+j)
  endfor
  for d=1 to D
      vid=0
  endfor
  improvei=0
end
```

Figure1. Pseudocode of SA_local_search

E. The Framework of GSMPSO

Based on the discussion above, the GSMPSO can be presented as a combination of the *lbest* with the Gaussian mutation operator and the *SA_local_search*. The scheme used in GSMPSO can described as:

Scheme 1: *SA_local_search* is only applied to the promising particle in the population, and the Gaussian mutation operator should be applied to all the stagnant particles in the population.

Although in some other MAs the local search is applied to all particles, we adopted **scheme 1** in the GSMPSO because the cost of local search is expensive and the frequent application of local search will result in a disastrous loss in diversity. In **scheme 1**, the population evolve along with the local refinement around the promising regions and the

mutation of the stagnant particles. A balance between the exploration and exploitation can be obtained in a time-saving way. A pseudocode for GSMPSO is described in Fig. 2.

IV. KNN REGRESSION BASED ON THE MEMETIC PSO

In this section, the KNN regression based on the weighted distance is studied and GSMPSO is used to train the weights of features for KNN regression. Each particle in GSMPSO represents an assignment of weights to all the features and the fitness of a particle is determined by the mean squared errors (MSE) of the training example set.

```
for each particle i
   initialize each particle
end
do
   for each particle i
     calculate particle velocity v_i according equation (1)
     update particle position z_i according equation (2)
     if the f(x_i)< f(pbest_i)
     then pbest_i =x_i
       improve_i++
       no_ improve_i= 0
     else
       no_ improve_i
       improve_i = 0
     endif
   endfor
   for each particle i
     if i is stagnant
     then mutate i according to (5)
     else if i is promising
     then call the procedure SA_local_search(i) (Fig.2)
     endif
   endfor
   choose the particle with the minimum f(pbest_i)
   of all the particles as the gbest
while maximum iterations is not attained
output the f(gbest)
```

Figure2. Pseudocode of GSMPSO

A. K Nearest Neighbours Regression

KNN regression [15] is an instance-based learning or lazy learning. For an object with unknown value, the weighted average of the values of one object's k nearest neighbours is assigned to the object. The training examples are vectors in a D-dimension feature space, each with a property value. In the training phase, feature vectors are stored as the training examples. In the regression phase, for the object e_i with unknown value, k nearest neighbours $e_{i1},\ldots,e_{ik}$ is selected from the training examples according to the similarity measurement. The similarity is usually measured by the Euclidean distance. In this work, a weighted Euclidean distance (8) [16] is used. The weighted average of the values of e_i's k nearest neighbours is computed by (7) in which $w_{ij}=1/d(e_i,e_{ij})$. $\hat{f}(e_i)$ is the approximation of $f(e_i)$.

$$\hat{f}(x_i)=\frac{\sum_{j=1}^{k} w_j f(e_{ij})}{\sum_{j=1}^{k} w_j} \tag{7}$$

$$d(x_i,x_j)=\sqrt{\sum_{k=1}^{D} w_j (x_{ik}-x_{jk})^2} \tag{8}$$

B. Coding of the Particle

We used the coding mechanism in [17]. The particle x_i represents a D dimension vector in which x_{ij} represents the weight of the jth feature, $0\le x_{ij}\le 1$. Therefore a particle represents an assignment of weights to all features.

C. Objective Function

In GSMPSO-KNN regression, for a given training example set TE and a particle x_i, the fitness of x_i can be determined by the following steps:

1) For each example e_i in TE, find its k nearest neighbors $e_{i1},\ldots,e_{ik}$ from TE-$\{e_i\}$ by (8) in which the weight w_j is assigned by x_{ij} which is the jth value of the particle x_i. $\hat{f}(e_i)$ is obtained by (7).

2) The fitness of x_i is evaluated by the MSE of TE (9).

$$MSE(TE)=\sqrt{\frac{\sum_{e\in TE}(f(e)-\hat{f}(e))^2}{|TE|}} \tag{9}$$

In (9), we can found that in the training phase of GSMPSO-KNN regression, GSMPSO is used to adjust of the weights of features to fit the training examples.

V. EXPERIMENT RESULT

A. Parameter Setting

In this experiment, the GSMPSO is compared with the original PSO on training the weights of features for KNN regression. All the common parameters are set as follows: k=10, the population size is set to 10 and the maximum number of iterations is set to 1000. The constriction factor is set to 0.729. $c_1$and c_2 are both set to 2.05. The range of the velocity vector is set as the same as the range of the problem. The parameters used in *SA_local_search* were set as follows: T_0=10, c_1=1.5, *ls_length*=100, w decreases from 0.9 to 0.4 linearly along with the evolution cycles. Each PSO is run for 100 times.

B. Result and Discussion

The training example set obtained by simulation in section II is fed into PSO-KNN regression and GSMPSO regression to train the weights of the features. After the training phase, the KNNs trained by the PSOs are applied to the test data set and a prediction performance can be obtained. The GSMPSO-KNN regression is compared with the PSO-KNN regression and the original KNN regression based on the basic Euclidean distance. The result in Table I shows that compared with the PSO-KNN and KNN, the GSMPSO can fit the training examples better and a more accurate result on the

test examples is also acquired. The effectiveness of the memetic strategy is demonstrated.

TABLE I RESULT OF THE KNN MEHTODS

Result / Algorithm	Training MSE	Prediction MSE
KNN	215.489±0.0	113.714±0.0
PSO-KNN	172.134±3.1186	79.454±7.077
GSMPSO-KNN	169.667±1.765	75.724±4.208

In Table II, we list the weights of the features trained by GSMPSO. These features are defined in section II. We can found that in this experiment, the attributes of the lot is more important than the attributes describing the status of the fab because the production model is simulated under a moderate loading environment. The redundant attributes can also be detected. For example, the QBT_t is a redundant attribute because it is highly relevant to QF_t and QBN_t. SK_i holds a high weight because the rules of thumb used by the fab staffs is due-date related, this is also consistent with the assumption.

TABLE II WEIGHTS OF THE FEATURES

Feature	WBN_t	WBT_t	QF_t	QBN_t	QBT_t
Weight	0.0	0.119	0.109	0. 137	0.0
Feature	NRB_i	NOP_i	SKi	TP_i	NSL_i
Weight	1.0	1.0	1.0	0.35	0.916

To further demonstrate the proposed method. The training example set is also fed into some ANNs including Radial Basis Function (RBF) ANN, Back Propagation (BP) ANN and Generalized ANN implemented in MATLAB 2008b. The result can be found in Table III. By the ANN methods, the training algorithm can reach a smaller MSE to fit the training example set but it is easy for ANNs to overfit the training examples and perform badly on the test data set. So in this experiment, the prediction MSE of ANNs is even worse than the original KNN method. But we cannot conclude that the KNN methods outperform the ANN methods because the training examples are generated by the simulation under a stable manufacturing environment. And the data generated by simulation is smooth and normal. The advantages including tolerance of noises, capability of simulating complex systems of ANNs may work under other manufacturing environments or on the data from the real fab.

TABLE III RESULT OF THE ANN MEHTODS

Result / ANN	Training MSE	Prediction MSE
RBF	77.959	150.632
BP	65.657	168.804
GRNN	71.993	134.102

VI. CONCLUSION AND FUTURE WORK

In this work, we study the problem of cycle time prediction of wafer lots. A novel memetic algorithm GSMPSO is proposed to weight the features for KNN regression and the GSMPSO-KNN regression method is adopted to predict the cycle time of wafer lots. The experiment result demonstrates that GSMPSO-KNN regression outperforms the original PSO-KNN regression on fitting the training example set and a more accurate prediction result can also be obtained. Compared with some ANNs, GSMPSO-KNN regression is still outstanding for its accuracy and the training result of GSMPSO-KNN regression is more comprehensible to the fab staffs. In the future work, we will study the cycle time prediction problem under an unstable and fluctuated manufacturing environment. Some other factors should also be taken into account. Furthermore, a practical application of the GSMPSO-KNN regression to the real data from the fab is meaningful.

ACKNOWLEDGMENT

This paper is supported in part by National Natural Science Foundation of China (No.50905129, No. 61034004), Natural Science Foundation of Shanghai (No. 11ZR1440400), the Program for Scientific Innovation of Shanghai (No.09DZ1120600, 10DZ1120100) and Shanghai Leading Academic Discipline Project (No. B004).

REFERENCES

[1] Baker KR, Sequencing rules and due-date assignment in a job shop, Management Science, vol.30, no.9, pp.1093–1104, 1984.
[2] Baker KR, Bertrand JWM, A comparison of due-date selection rules, AIIE Trans, vol.13, no.2, pp.123–131, 1981.
[3] Chang F CR, A study of factors affecting due-date predictability in a simulated dynamic job shop. Journal of Manufacturing Systems, vol.13, no.6, pp.393–400, 1994.
[4] Barman, The impact of priority rule combinations on lateness and tardiness, IIE Transactions, vol.30, no.5, pp.495-504, 1998.
[5] Chen T. A fuzzy back propagation network for output time prediction in a wafer fab, Applied Soft Computing, vol.2, no.3, pp.211–222, 2003.
[6] Chen T. A Hybrid SOM-BPN Approach to Lot Output Time Prediction in a Wafer Fab. Neural Processing Letters, vol.24, no.4, pp.271–288, 2006.
[7] J. Kennedy, R. C. Eberhart, Particle swarm optimization, In Proceedings of IEEE International Conference on Neural Networks, IV, pp.1942-1948, 1995.
[8] J. C. Ni, L. Li, F. Qiao, Q. D. Wu, "A Novel Memetic Algorithm and its Application to Data Clustering," unpublished.
[9] N. Krasnogor and J. E. Smith, A tutorial for competent memetic algorithms: Model, taxonomy and design issues. IEEE Trans. Evol. Comput., vol.9,no.5,pp.474-488, 2005.
[10] H. Higashi, H. Iba, Particles warm optimization with Gaussian mutation. In Proceedings of the IEEE Swarm Intelligence Symphosium, pp.72-79, 2003.
[11] P.S. Andrews. An investigation into mutation operators for particle swarm optimization. In Proceedings of the Congress on Evolutionary Computation, pp. 1044-1051, 2006.
[12] J. Kennedy, Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, In Proceedings of the Congress on Evolutionary Computation, pp. 1931-1938, 1999
[13] J. Kennedy, The particle swarm: social adaptation of knowledge Proceedings of IEEE International Conference on Evolutionary Computation, pp.303-308, 1997.
[14] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi. Optimization by simulated annealing. Science, vol.220, pp.671-680, 1983.
[15] L X Jiang, Z H Cai, D H Wang, S W Jiang. Survey of improving k-nearest-neighbor for classification. International conference on Fuzzy Systems and Knowledge Discovery, pp. 679-683, 2007.
[16] J D Kelly. A Hybrid Genetic Algorithm for Classification, Proceedings of the 12th international joint conference on Artificial intelligence, pp.645-650, 1991.
[17] J T Ren, X L Zhuo, X L Xu, S C Yin. PSO Based Feature Weighting Algorithm for KNN, vol.34, no.5, Computer Science, 2007 (in Chinese).