Melt Index Prediction by RBF Neural Network Optimized with an Adaptive New Ant Colony Optimization Algorithm

Jiubao Li, Xinggao Liu

State Key Laboratory of Industrial Control Technology, Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, People's Republic of China

Received 29 April 2010; accepted 13 July 2010
DOI 10.1002/app.33060
Published online 22 September 2010 in Wiley Online Library (wileyonlinelibrary.com).

ABSTRACT: Melt index (MI) is a crucial indicator in determining the product specifications and grades of polypropylene (PP). The prediction of MI, which is important in quality control of the PP polymerization process, is studied in this work. Based on RBF (radial basis function) neural network, a soft-sensor model (RBF model) of the PP process is developed to infer the MI of PP from a bunch of process variables. Considering that the PP process is too complicated for the RBF neural network with a general set of parameters, a new ant colony optimization (ACO) algorithm, *N*-ACO, and its adaptive version, A-*N*-ACO, which aim at continuous optimizing problems are proposed to optimize the structure parameters of the RBF neural network, respectively, and the structure-best models, *N*-ACO-RBF model and A-*N*-ACO-RBF model for the MI prediction of propylene polymerization process, are presented then. Based on the data from a real PP production plant, a detailed comparison research among the models is carried out. The research results confirm the prediction accuracy of the models and also prove the effectiveness of proposed *N*-ACO and A-*N*-ACO optimization approaches in solving continuous optimizing problem. © 2010 Wiley Periodicals, Inc. J Appl Polym Sci 119: 3093–3100, 2011

Key words: polypropylene; melt; computer modeling; adaptive ACO

INTRODUCTION

Production of polypropylene (PP) is a multibillion business, which has great influences on the world in the aspects of industry, military, economy, and so on. Melt index (MI) of PP produced, which is defined as the mass rate of extrusion flow through a specified capillary under prescribed condition of temperature and pressure,[1] is the most important parameter in determining the product's grade and quality control of practical industrial process. But it is usually measured offline in the laboratory with an analytical procedure, which is not only costly but also time consuming, leading to off-specification products and resulting in enormous losses in profit. An MI online estimation model is thereby very useful both as an online sensor and as a forecasting system.

Because MI is difficult to be measured directly, it is common to figure it out through an indirect method. Clearly, there are some certain relationship between the MI and some other easily measured variables, and the relationship can be used to develop the MI prediction model. To infer the difficult-measured variable from easy-measured variables, the chemical and physical relationships can be used; thus, an online analyzer can be constructed with the mechanism of polymerization process. However, the approach[2–6] is a big challenge because of the engineering activity and the relatively high complexity of kinetic behavior and operation of polymer plants as shown in Figure 1. The chemical and physical reactions in the reactors are so complicated that it is very difficult to model the reactors or the reaction processes happen in the reactors.[7,8] Even simplified mechanical models need to be developed with great effort to fit the inner relationship between MI and some of the factors in the reaction.[9–11]

To avoid the hardness of developing mechanism model, the empirical models based on data, and statistics provide an alternative to estimate the MI from the easy-measured variables without considering the complex chemical or physical reactions. Some production plants have used statistical methodologies to

Correspondence to: X. Liu (liuxg@iipc.zju.edu.cn).
Contract grant sponsor: National Natural Science Foundation of China; contract grant number: 50876093.
Contract grant sponsor: International Cooperation and Exchange Project of Science and Technology Department of Zhejiang Province; contract grant number: 2009C34008.
Contract grant sponsor: National High Technology Research and Development Program 863; contract grant number: 2006AA05Z226.
Contract grant sponsor: Science Fund for Distinguished Young Scholars of Zhejiang University; contract grant number: 581645.

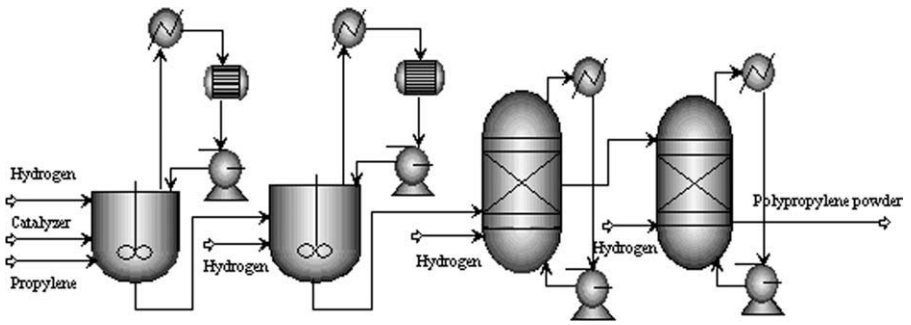

Figure 1 General scheme of propylene polymerization.

provide information for product and process design, monitoring, and control,[12–15] and, some researchers have strived to obtain easy empirical models through various methods. Han[16] used three different approaches, which are supported vector machines (SVM), partial least squares, and artificial neural networks, for MI estimation of SAN (styrene-acrylonitrile) and PP process. Shi[17,18] developed soft-sensor model for MI prediction based on weighted least squares support vector machines (LS-SVM) and independent component analysis, multiscale analysis and RBF. Neural networks have been widely applied to data-based model and control dynamic processes because of their extremely powerful adaptive capabilities in response to nonlinear behaviors.[19,20] Thus, Zhang[21] sequentially trained a set of neural networks, based on which the novel bootstrap aggregated neural networks are formed, and he obtained quite a good performance in the inferential estimation of the polymer MI in an industrial plant with the model developed by the aggregated neural networks. These works give good prediction, but greater performance and better universality of the estimation model are still the first-line goal in academic and industrial community.

In this work, the RBF neural network is used to fit the relationship between MI and a group of other easily obtained process variables in propylene polymerization, and then the MI prediction model, which can infer the MI of PP product from the batch of variables above, is developed. The key determining the RBF neural network's adaptive capability in response to the nonlinear relationship between MI and the chosen group of variables is the structure of the network, such as weights, centers, and bias. A general set of structure parameters do not guarantee the neural network to achieve the desired performance in the high complicated and nonlinear propylene polymerization process. Therefore, an optimization of the neural network's structure parameters is necessary, and the work is carried out to improve the performance of the RBF neural network in MI prediction here.

In published literature, lots of stochastic algorithms have been proposed to handle various academic and application optimizing problems like the one mentioned earlier. particle swarm optimization, ant colony optimization (ACO), genetic algorithm (GA), and simulated annealing are examples of some popular intelligent algorithms. Among these algorithms, ACO is a novel nature-inspired heuristic method for the solution of combination problems, which only involves various basic mathematic operations and just needs the value of objective function exports without gradient information. It shows great search capacity for optimum and thus is used here to optimize the structure parameters of RBF neural network. The parameters to be optimized here are continuous values, but traditional ACO approach to handle this problem is discrete the continuous problem, which results in the lack of accuracy for the solution it obtains. Hence, a novel ACO algorithm, *N*-ACO algorithm, which is designed for continuous optimizing problem, is proposed to optimize the structure of the RBF neural network and the structure-best neural network for the MI prediction model, *N*-ACO-RBF model, is obtained. Considering that a defect lies in the *N*-ACO algorithm, an adaptive *N*-ACO, A-*N*-ACO, is further proposed, and an A-*N*-ACO-RBF model is thereby further developed.

So far, there is little published work on how to optimize the RBF neural network with intelligent stochastic algorithms for the prediction of PP MI. Here, an innovational trial based on the thought is carried out, and, finally, the newly MI prediction models, *N*-ACO-RBF model and A-*N*-ACO-RBF model for PP polymerization process are achieved. The performance of the proposed models is illustrated and evaluated based on some real industrial processing data. The results obtained are then discussed, and concluding remarks about the design are presented.

RBF NEURAL NETWORK AND ACO ALGORITHMS

RBF neural network

The RBF neural network is the primary choice for process product quality prediction most times,

because any of the industrial processes is characterized by strong nonlinear and correlated relationships, whereas the RBF neural network provides great global approximation and convergence in fitting these complex relationships.[22,23] It is a typical feed-forward network, which takes a structure of three layers: the input layer, the hidden layer, and the output layer. The input layer collects the input information and formulates the input vector x. The hidden layer is composed by L hidden nodes, which apply nonlinear transformations to the input vector. The output layer gives the final responses. The RBF neural network can be considered as a mapping in Euclidean space: $T : R^r \rightarrow R^s$. Let $x^p \in R^r$ be the input vector and $c^i \in R^r (i = 1, 2, \ldots, k)$ be the center. The output is formed by a linear combination of the hidden layer responses, given by

$$y_j(x^p) = \sum_{i=1}^{k} w_{ji} \Phi_i(\|x^p - c^i\|), j = 1, 2, \ldots, s \quad (1)$$

where $\|\cdot\|$ is the Euclidean distance, k is number of the hidden layer nodes, $\Phi_i(\cdot)$ is hidden layer node response, w_{ji} is the output weight, x^p is the input vector, y_j is the output of jth output node, and s is the number of the output nodes. In the current model, a hidden layer node uses the Gaussian activation function to make a response, that is

$$\Phi_i(\|x^p - c^i\|) = \exp\left(-\frac{(\|x^p - c^i\|)^2}{2\sigma_i}\right), i = 1, 2, \ldots, k \quad (2)$$

where c^i and σ_i are the center and the width of the ith node in the hidden layer respectively. In this work, the center c^i is obtained by the orthogonal least square algorithm,[24] and the width σ_i is determined by experience and they determine the receptive field around the node.

Basic ACO algorithm

Ant colony optimization (ACO) algorithms are some kind of heuristic search techniques, which are inspired by the foraging mechanism of a real ant system and especially by the ability of the ants to figure out the shortest path between their nests and the food source.[25] When searching for food, ants track a volatile chemical pheromone, which is released by ants, to choose the paths to head, and they prefer the paths marked by high-pheromone concentration. In this way, ants can communicate with one another through an indirect mechanism based on the modification of pheromone concentration by the physical environment and the ant individuals.

At the beginning of the search process, there is no pheromone, and ants choose their paths randomly, releasing pheromone along the followed path. After a while, some paths between nests and food source are figured out, and the shorter ones take high-pheromone concentration, because walking through a shorter path costs less time and during a certain period of time more ants will have passed this shorter path and left more pheromone. However, ants take the possibility to abandon the high-pheromone concentration path and head to other paths, which will result in the discovery of better and new paths. In the long run, with the presence of many pheromone trails, ants' pheromone-driven behavior makes the strong concentration path the most promising one and thus forms a positive-feed-back process to converge to the shortest path between nests and food source.[26]

ACO algorithm simulates the foraging process of ant system and was first mathematically proposed in early 1990s by Dorigo.[27] ACO requires a problem to be represented as a graph consisting of several nodes and edges, and these edges can be formed to a route, which represents a solution to the problem.[28,29] Let $a_i(t)$ be the number of ants in node n_i at time t, $\tau_{ij}(t)$ is the pheromone concentration in the edge e_{ij} between node n_i and n_j at time t, n is the total number of the nodes, and m is the total number of ants. N is the set of nodes and E is the set of edges. $\Gamma = \{\tau_{ij}(t) | e_{ij} \in E\}$ is the set of pheromone concentration of all the edge at time t. ACO search for the best path consisting of several edges according to the following steps:

Step 1. Initialize the graph by setting $\tau_{ij}(t)$ to a constant $\tau_{ij}(0)$ and deciding a group of integers $a_i(0)$ satisfying with: $m = \sum a_i(0)$ and n_i here have $a_i(0)$ ants in must be an admissible starting node. Also initialize the sequence number of iteration: $k = 1$.

Step 2. Initialize the sequence number of ant: $b = 1$.

Step 3. For ant b, choose a sequence of edges to form a route $r_b(k)$, which can represent a solution to the optimizing problem. $p_{ij}^b(k)$ expresses the probability for ant b to choose the edge e_{ij} from node i in iteration k, and it is given by

$$p_{ij}^b(k) = \begin{cases} \dfrac{\tau_{ij}(k)}{\sum_{s \subset \text{allowed}_b} \tau_{is}(k)}, if & j \in \text{allowed}_b \\ 0, if & j \notin \text{allowed}_b \end{cases} \quad (3)$$

The expression allowed_k indicates the admissible edges from node i for ant b.

Step 4. Calculate the fitness $L_b(k)$ of the route $r_b(k)$ according to the objective function of the optimizing problem.

Step 5. $b = b + 1$, if $b > m$ go to Step 6; else, go back to Step 3.

Step 6. Update the pheromone concentration of all the edges in the graph according to the following rule:

$$\tau_{ij}(k+1) = (1-\rho)\tau_{ij}(k) + \Delta\tau_{ij}(k) \quad (4)$$

where $\rho(0 \leq \rho \leq 1)$ is a coefficient of pheromone evaporation, $\Delta\tau_{ij}(k)$ is the total adjustment of pheromone concentration on edge e_{ij}, which is obtained by

$$\Delta\tau_{ij}(k) = \sum_{b=1}^{m} \Delta\tau_{ij}^{b}(k) \quad (5)$$

$$\Delta\tau_{ij}^{b}(k) = \begin{cases} \frac{L_b(k)}{Q}, if & e_{ij} \in r_b(k) \\ 0, if & e_{ij} \notin r_b(k) \end{cases} \quad (6)$$

where m is the total number of ants, $\Delta\tau_{ij}^{b}$ (k) is the adjustment resulted by the completing of route $r_b(k)$ of ant b in iteration k, and Q is an adjustable parameter.

Step 7. $k = k + 1$, if $k >$ iter$_{max}$, go to Step 8; else, go back to Step 2. Here, iter$_{max}$ is the given max number of iteration.

Step 8. Choose the route with the best fitness that has been searched so far as the final solution to the optimizing problem.

New ACO algorithm (*N*-ACO)

As introduced earlier, basic ACO requires the problem to be presented as a discrete one.[30] Thus, when coming to a continuous optimizing problem, the solution space should be cut into a grid space first. Take a D-dimensional problem for example, the ith dimension should be a real number in the zone $[\min_i, \max_i]$, then, in ACO, the zone is cut into several intervals, and every interval is represented as a node. An ant choosing a node means choosing a random number in the specific interval. This method of handling the continuous problem cannot assure the accuracy of solution, because the zone that has been cut can be a very large interval and a node is mapped to a big interval while ACO only chooses a random number in this interval. To get accurate solution, the zone should be transferred to many more intervals, which will result in a time-consuming situation in ACO search.

Here, a new ACO algorithm (*N*-ACO), which is designed for continuous optimizing problems, is presented. A solution is not represented as a route consisting of several nodes, but just one node that is vector. The nodes represent food sources with certain pheromone concentrations, which are related to the quantity of food in these sources. The ants' objective is to find out the food source with most quantity of food. An ant starts searching with choosing a food source by a pheromone concentration-related probability and tries to search around the food source it has chosen to find a more attractive food source. If a better food source, which means it contains more food than the original source does, is obtained, the ant gets excited, and releases more pheromone according to the quantity of the food found. From mathematical point of view, food source is a solution to the optimizing problem, and the quantity of food in the source is the fitness of the solution. Ants tend to search around good food sources to find out even better sources, and it means that the algorithm is inclined to look for better solutions of the problem around the existing solutions with high fitness. It is the mechanism of the ants' "local search," which will be introduced later.

Sometimes, the food quantity in some sources is small and, at the same time, starting from these points, ants cannot find out better food sources than the original ones. Then, these food sources should be replaced with some others, which are more potential to have wonderful food sources around. To form the new food sources, the replaced ones are used to make combinations, where the ideas of mutation and cross basically used in GA algorithm are used.[31] The thinking above is the ants' "global search" and the details will be explained subsequently.

The *N*-ACO algorithm is carried out as follows:

Step 1. Prepare for the algorithm:

1.1. Initialize the searching space by giving the total number of ants m and a series of initial food sources $S = (s_1, s_2,\ldots,s_n)$. A food source $s_i = (x_1, x_2,\ldots x_D)$ $(i = 1, 2,\ldots n)$ represents a solution to the D-dimension continuous optimizing problem and n is the total number of starting points that ants will search around.

1.2. Calculate and record the fitness of $(s_i = 1, 2,\ldots n)$ as $(F_i = 1, 2,\ldots,n)$.

1.3. Initialize the sequence number of iteration $k = 1$.

1.4. Specify the parameters of global search: R_1 and R_2. Here, R_1 is the number of sources that should be replaced by the new ones created with mutation operation, and R_2 is the number of sources that should be replaced by the new ones created with cross operation.

Step 2. Initialize the sequence number of ant $j = 1$ and do the local search:

2.1. Calculate the probability for choosing the source $(s_i = 1, 2,\ldots,n)$ by:

$$P_i(k) = \frac{F_i}{\sum_{i=1}^{n} F_i} (i = 1, 2,\ldots, n) \quad (7)$$

2.2. Choose a s_{chosen} by following the roulette rules for ant j. Meanwhile, make sure that every source not be chosen more than once during the kth iteration.

2.3. Create a distance with a direction del $= (d_1, d_2,\ldots,d_D)$ that ant j will walk, and a new source s_{new} is obtained by:

$$s_{new} = s_{chosen} + \text{del} \quad (8)$$

Calculate the fitness of s_{new} and save as F_{new}, if $F_{new} > F_{chosen}$, s_{new} will be accept and replace s_{chosen} in the set S. F_{chosen} in the set $F_i(i = 1,2,\ldots,n)$ is also updated. Else, if $F_{new} \leq F_{chosen}$, do nothing.

2.4. $j = j + 1$; if $j > m$, go to Step 3; else go back to 2.1.

Step 3. In Step 2, local search has updated the set S as well as $F_i(i = 1,2,\ldots,n)$, and here the global search is processed:

3.1. Choose the sources to be replaced by following the roulette rules. The probability for source $s_i(i = 1,2,\ldots,n)$ to be chosen is given by:

$$Q_i(k) = \frac{1/F_i}{\sum_{i=1}^{n} 1/F_i} (i = 1, 2, ..., n) \quad (9)$$

The total number of sources to be chosen is $R_1 + R_2$.

3.2. For the first R_1 sources to be replaced with mutation operation, let each initial source s_{old} shift a random distance in a random direction del = (d_1, d_2,...,d_D) to form a new source s_{new}:

$$s_{new} = s_{old} + \text{del} \quad (10)$$

Replace the chosen R_1 sources with the new ones.

3.3. For the left R_2 sources to be replaced with cross operation, let the initial source s_{old} cross with a random source s_{random} in S to get a new one s_{new}:

$$s_{new} = p \cdot s_{old} + (1 - p) \cdot s_{random} \quad (11)$$

where p is a probability parameter that can be adjusted.

Replace the chosen R_2 sources with the new ones.

Step 4. $k = k + 1$; if $k > \text{iter}_{max}$, go to Step 5; else go back to Step 2.

Step 5. Take the source s_{bet} with best fitness F_{best} to be the final solution of the optimizing problem.

Adaptive *N*-ACO algorithm

Although the *N*-ACO has greatly improved the effectiveness and the capacity of basic ACO in handling continuous optimizing problems, there is a vital defect lies in Step 2 (substep 2.3), which largely influences the convergence of the *N*-ACO algorithm and the reason is simple. When proceed local search according to this strategy, there are no certain directions and distances leading the ants to better food sources, but all random created distances in random directions. It results the difficulty for ants to find out the better sources, and in the anaphase of the *N*-ACO, ants can hardly get the exact optimum of the problem around the local best solutions, which means a poor convergence of the *N*-ACO.

In basic ACO algorithm, there are lots of adaptive ways[32–34] to improve its performance, but here for the *N*-ACO, they can hardly be used. Instead, based on the following three ideas, an adaptive modification of the *N*-ACO is proposed to overcome the defect mentioned earlier. First, an ant will not just try once around a chosen source, but several times to make full use of the experience it obtains from formal tries. Second, when an ant finds a better source with a distance in a direction, it moves in the same direction with the same distance in its next try, regarding this distance in the direction as a heuristic one. Third, if an ant fails in any try, the direction changes and the distance is decreased according to its number of tries. At the same time, the distance also decreases according to the iteration number, which contributes a lot the convergence of the algorithm. Overview, it makes the distance and the direction adaptive to the number of iteration and times of the ant's try. Therefore, the *N*-ACO algorithm is modified to be an adaptive one, A-*N*-ACO algorithm, and its procedure differs with *N*-ACO only in Step 2 (substep 2.3) as following:

Step 2. The same **2.1**, **2.2** and **2.4** as that in N-ACO.

2.3. Adaptive steps of A-*N*-ACO:

2.3.1. Initialize the number of try times for ant j around s_{chosen} : $q = 1$.

2.3.2. Create an iteration number related (The larger iteration number, the smaller distance) distance with a direction del = (d_1, d_2,...,d_D) that the ant j will walk, and a new source s_{new} is obtained by:

$$s_{new} = s_{chosen} + \text{del} \quad (12)$$

Calculate the fitness of s_{new} and save as F_{new}, if $F_{new} > F_{chosen}$, s_{new} will be accept and replace s_{chosen} to be the new starting point for the next search try of ant j. Keep the del = (d_1, d_2,...,d_D) unchanged.

Else, if $F_{new} \leq F_{chosen}$, do nothing but create a new del = (d_1, d_2,...,d_D) in a different direction and with a decreased distance.

2.3.3. $q = q + 1$, if $q > q_{max}$, replace the original s_{chosen} with the finally searched source s_{new}, updated the related F_{chosen} and go to 2.4. Else, go back to 2.3.2.

RESULTS

The process considered here is a propylene polymerization process, which is currently operated for commercial purposes in a real plant in China. A highly simplified schematic diagram of this process is illustrated in Figure 1. The process consists of a chain of reactors in series, two continuous stirred-

tank reactors (CSTR) and two fluidized-bed reactors (FBR). Hydrogen is fed into each reactor, but the catalyst and propylene are added only to the first reactor along with the solvent. These liquids and gases supply reactants for the growing polymer particles and provide the heat transfer media. The polymerization reaction takes place in a liquid phase in the first two reactors and is completed in vapor phase in the third and fourth reactors to produce the powdered polymer products. The MI of the PP, which determines the quality of the product, depends on the catalyst properties, reactant composition, and reactor temperature. Hydrogen regulates the molecular weight of PP.

Therefore, a pool of process information formed by nine process variables (T, p, l, a, f_1, f_2, f_3, f_4, and f_5), which influence the process greatly, have been chosen to develop the MI prediction model. T, p, l, and a stand for process temperature, pressure, level of liquid, and percentage of hydrogen in vapor phase in the first CSTR reactor, respectively. f_1, f_2, and f_3 are flow rate of three streams of propylene into the first CSTR reactor, and f_4 and f_5 are flow rate of catalyst and cocatalyst into the first CSTR reactor. The data have been acquired from the historical log recorded in a real propylene polymerization plant, and they are filtered to discard abnormal situations and to improve the quality of prediction model. The input and output variables are also normalized with respect to their maximum operation values. The selection of the most suitable training dataset among all the available process information is one of the most important tasks in model learning. The method to construct the training dataset here is selecting data according to the time series of recorded data, then RBF model, *N*-ACO-RBF model, and A-*N*-ACO-RBF model are developed based on the selected training dataset. All the data are separated into training dataset, testing dataset, and generalization dataset, the latter two of which are used to test the accuracy and the universality of the models. There are 120 points in the training dataset, 30 points in the testing dataset respectively, and the left are all in the generalization dataset. It is noted that the testing dataset and training dataset are from the same batch, whereas the generalization dataset is derived from another batch.

To see about the MI prediction accuracy of models, the difference between the output of the models and the desired output (the analytic MI values from laboratory) is considered as the error and represented in several ways. In this work, the following measures are used for model evaluations: the mean absolute error (MAE), the mean relative error (MRE), the root of mean square error (RMSE), and the Theil's inequality coefficient (TIC). The error indicators are defined as following:

TABLE I
Performance of the Models on the Testing Dataset

Model	MAE	MRE (%)	RMSE	TIC
RBF	0.0278	1.06	0.0354	0.0068
N-ACO-RBF	0.0203	0.77	0.0261	0.0050
A-N-ACO-RBF	0.0115	0.44	0.0153	0.0029

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{13}$$

$$\mathrm{MRE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{14}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{15}$$

$$\mathrm{TIC} = \frac{\sqrt{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}}{\sqrt{\sum_{i=1}^{N} y_i^2} + \sqrt{\sum_{i=1}^{N} \hat{y}_i^2}} \tag{16}$$

where y_i and $\hat{y}_i$ denote the measured value and predicted result, respectively.

The data listed in Table I show that the A-*N*-ACO-RBF model functions best overall on the testing dataset. In details, RBF model gives an MAE of 0.0278, an MRE of 1.06%, a RMSE of 0.0354, and a TIC of 0.0068. For *N*-ACO-RBF model, an optimized RBF model by the *N*-ACO algorithm, obtains an MAE of 0.0203, an MRE of 0.77%, a RMSE of 0.0261, and a TIC of 0.0050. *N*-ACO-RBF model has already obtained improved prediction accuracy than the RBF model. However, the A-*N*-ACO-RBF model achieves even better performance. The MAE, MRE, RMSE, and TIC are 0.0115%, 0.44%, 0.0153%, and 0.0029%, with percentage decreases of 58.63%, 58.49%, 56.78%, and 57.35% compared to that of the RBF model, respectively. These error indicators prove the A-*N*-ACO-RBF model provides wonderful MI prediction accuracy for the propylene polymerization process.

Figure 2 gives a more explicit illustration in how better the A-*N*-ACO-RBF works than RBF model and *N*-ACO-RBF model do on the testing dataset. The curve without mark is the real MI value obtained from analysis in laboratory, whereas the curve marked with squares is the MI value predicted by RBF model. The results predicted by *N*-ACO-RBF model and A-*N*-ACO-RBF model are represented by the curves marked with circles and crosses, respectively. Obviously, the A-*N*-ACO-RBF model's result is best and nearly being the real MI value on most data points. The *N*-ACO-RBF model's prediction is better than RBF model, but not as good as that of the A-*N*-ACO-RBF model. The visual comparison proves the great prediction accuracy of A-*N*-ACO-RBF model.

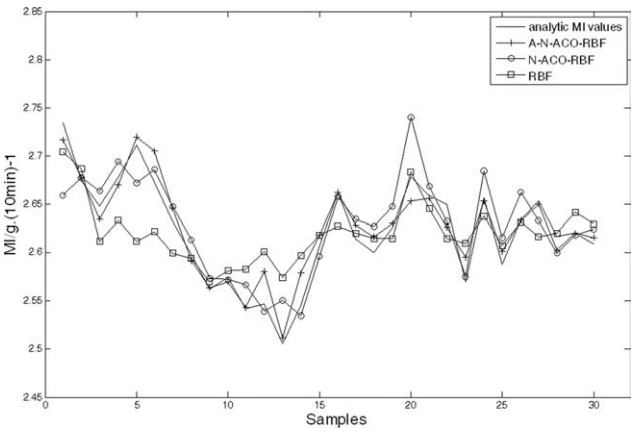

Figure 2 Performance of the models on the testing dataset.

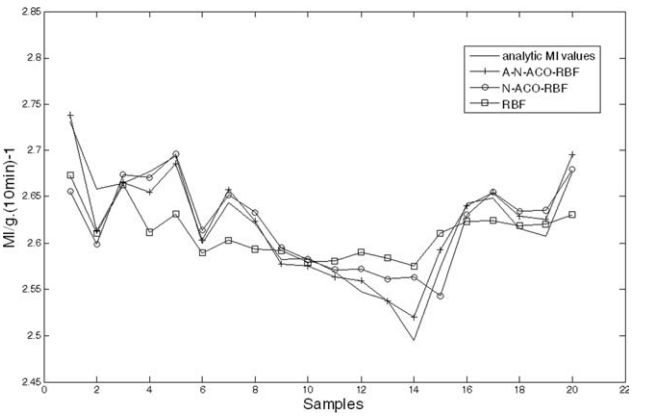

Figure 3 Performance of the models on the generalization dataset.

To specify the universality of the proposed MI prediction models, models are also evaluated on the generalization dataset. An accurate prediction of MI on this dataset gives a strong support that the model owns good universality.

Table II lists the specific error indexes for RBF model *N*-ACO-RBF model and A-*N*-ACO-RBF model when they predict on generalization dataset. Here, A-*N*-ACO-RBF model cuts down the other two models again, with a decrease of 63.71% in MRE from 1.24% to 0.45%, compared to the RBF model. And almost the same things happen in terms of MAE, RMSE, and TIC. N_ACO-RBF model still win over RBF model, but loose to A-*N*-ACO-RBF model.

At the same time, another comparison in how the models work on generalization dataset is given in Figure 3, which speaks even more powerfully than the data in Table II does. Here, the curves marked with squares, circles, and crosses are still the MI values predicted by RBF model, *N*-ACO-RBF model, and A-*N*-ACO-RBF model, respectively, while the analytic MI value curve is without mark. Clearly, A-*N*-ACO-RBF model gives a nearly real MI value prediction, much more accuracy than RBF model and *N*-ACO-RBF model do. Thus, it is proved that the A-*N*-ACO-RBF model holds excellent universality in MI prediction both statistically and graphically.

Furthermore, the method proposed by Shi[17] has obtained an MAE of 0.0635 for MI prediction on testing dataset. But the A-*N*-ACO-RBF model here achieved an MAE of 0.0115, with an obviously huge percentage decrease compared to the published result. It also strongly supports the proposed method here.

TABLE II
Performance of Models on the Generalization Dataset

Model	MAE	MRE (%)	RMSE	TIC
RBF	0.0326	1.24	0.0398	0.0076
N-ACO-RBF	0.0209	0.80	0.0300	0.0057
A-N-ACO-RBF	0.0118	0.45	0.0158	0.0030

CONCLUSION

An *N*-ACO algorithm and its adaptive version, A-*N*-ACO algorithm, which aim at continuous optimizing problems in MI prediction of propylene polymerization process are presented and used to search for the optimum of the RBF neural network's structure parameters in this work. Then, the *N*-ACO-RBF model and A-*N*-ACO-RBF model to estimate the MI of PP from other process variables are developed and evaluated on some data from a real industrial plant, compared to the RBF model without optimization. The RBF model has a worst performance, because it predicts with a slightly big error and the *N*-ACO-RBF model does much better. However, the A-*N*-ACO-RBF model achieves even better performance and predicts the MI with an MRE of 0.44%, decreases with a percentage of 58.49% and 42.86% when compared with whose of 1.06% and 0.77% obtained from the corresponding RBF and *N*-ACO-RBF models, respectively. The results obtained indicate that the proposed A-*N*-ACO-RBF model provides prediction accuracy and reliability and supposed to have a promising outlook in practical use.

References

1. Bafna, S. S.; Beall, A. M. J Appl Polym Sci 1997, 65, 277.
2. McAuley, K. B.; MacGregor, J. F. AIChE J 1991, 37, 825.
3. McKenna, T. F.; Soares, J. B. P. Chem Eng Sci 2001, 56, 3931.
4. Sarkar, P.; Gupta, S. K. Polym Eng Sci 1993, 33, 368.
5. Lee, E. H.; Kim, T. Y.; Yeo, Y. K. Kor J Chem Eng 2008, 25, 613.
6. Castoldi, M. T.; Pinto, J. C.; Melo, P. A. Ind Eng Chem Res 2007, 46, 1259.
7. Kanellopoulos, V.; Tsiliopoulou, E.; Dompazis, G.; Touloupides, V.; Kiparissides, C. Ind Eng Chem Res 2007, 46, 1928.

8. Khare, N. P.; Lucas, B.; Seavey, K. C.; Liu, Y. A.; Sirohi, A.; Ramanathan, S.; Lingard, S.; Song, Y. H.; Chen, C. C. Ind Eng Chem Res 2004, 43, 884.
9. Azizi, H.; Ghasemi, I.; Karrabi, Q. Polym Test 2008, 27, 548.
10. Ghasemi, S. M.; Sadeghi, G. M. M. J Appl Polym Sci 2008, 108, 2988.
11. Nele, M.; Latado, A.; Pinto, J. C. Macromol Mater Eng 2006, 291, 272.
12. Ahmed, F.; Nazir, S.; Yeo, Y. K. Kor J Chem Eng 2009, 26, 14.
13. Sharmin, R.; Sundararaj, U.; Shah, S.; Griend, L. V.; Sun, Y. J Chem Eng Sci 2006, 61, 6372.
14. Rallo, R.; Ferre-Gine, J.; Arenas, A.; Giralt, F. Comput Chem Eng 2002, 26, 1735.
15. Kaneko, H.; Arakawa, M.; Funatsu, K. AIChE J 2009, 55, 87.
16. Han, I. S.; Han, C.; Chung, C. B. J Appl Polym Sci 2005, 95, 967.
17. Shi, J.; Liu, X. G. J Appl Polym Sci 2006, 101, 285.
18. Shi, J.; Liu, X. G. Neurocomputing 2006, 70, 280.
19. Chaudhuri, B. B.; Bhattacharya, U. Neurocomputing 2000, 34, 11.
20. Hunt, K. J.; Sbarbaro, D.; Zbikowski, R.; Gawthrop, P. J Automatica 1992, 28, 1083.
21. Zhang, J.; Jin, Q. B.; Xu, Y. M. Chem Eng Technol 2006, 29, 442.
22. Park, J.; Sandberg, I. W. Neural Comput 1993, 5, 305.
23. Ahmad, S.; Simonovic, S. P. J Hydrol 2005, 315, 236.
24. Chen, S.; Cowan, C. F. N.; Grant, P. M. IEEE Trans Neural Network 1991, 2, 302.
25. Li, L.; Qiao, F.; Wu, Q. D. Int J Adv Manuf Technol 2009, 44, 985.
26. Aymerich, F.; Serra, M. Compos A Appl Sci Manuf 2008, 39, 262.
27. Ho, C. K.; Ewe, H. T. Appl Artif Intell 2009, 23, 570.
28. Kanan, H. R.; Faez, K. Appl Math Comput 2008, 205, 716.
29. Kaveh, A.; Azar, B. F.; Hadidi, A.; Sorochi, F. R.; Talatahari, S. J Constr Steel Res 2010, 66, 566.
30. Baskan, O.; Haldenbilen, S.; Ceylan, H.; Ceylan, H. Appl Math Comput 2009, 211, 75.
31. Lee, Z. J.; Su, S. F.; Chuang, C. C.; Liu, K. H. Appl Soft Comput 2008, 8, 55.
32. Yang, Y. J.; Wu, C. Expert Syst Appl 2009, 36, 3034.
33. Lin, Y.; Zhang, J.; Xiao, J. Appl Math Comput 2008, 205, 677.
34. Duan, H. B.; Zhang, X. Y.; Wu, J.; Ma, G. J. J Bionic Eng 2009, 6, 161.