Artificial Immune Systems in Industrial Applications

Dipankar Dasgupta
Dept. of Mathematical Sciences
The University of Memphis
Memphis, TN 38119

Stephanie Forrest
Dept. of Computer Science
The University of New Mexico
Albuquerque,NM 87131

Abstract

Artificial Immune System (AIS) is a new intelligent problem-solving technique that being used in some industrial applications. This paper presents such an immunity-based algorithm for tool breakage detection. The method is inspired by the negative-selection mechanism of the immune system, which is able to discriminate between the self (body elements) and the non-self (foreign pathogens). However, in our industrial application, the self is defined to be *normal* cutting operation and the non-self is any deviation beyond allowable variation of the cutting force. The proposed algorithm is illustrated with a simulation study of milling operations and the performance of the algorithm in detecting the occurrence of tool breakage is reported. The results show that the negative-selection algorithm detected tool breakage in all the test cases.

Introduction

Manufacturers are always looking for ways to improve productivity without compromising on quality of manufacturing processes. To this end, much attention has been directed towards automated manufacturing. In drilling or high-speed milling industries, on-line monitoring of the tool breakage is a key component in unmanned machining operations.

In most milling industries, a reliable and effective tool breakage detection technique is required to respond to unexpected tool failure [Altintas and Yellowley, 1989]. In particular, such a monitoring technique is necessary to prevent possible damage to the workpiece and the machine tool or to avoid production of defective parts and possible overloading of tools. The normal operation of a milling cutter is often characterized from the measurements of some parameters that are correlated with tool wear. It is essential to detect the occurrence of abnormal events as quickly as possible before any significant performance degradation results. This can be done by continuous monitoring of the system for changes from the normal behavior patterns.

Thus, a signal may be sent to the machine controller/operator for triggering an emergency stop of the machine and the tool changed. Several techniques have been suggested in the literature for monitoring tool breakage in different machining operations [Altintas and Yellowley, 1989]. Recent efforts include Time series analysis [Tansel and McLaughlin, 1993a], Artificial Intelligence (AI) techniques [Chryssolouris and Guillot, 1990], pattern recognition methods [Li and Wu, 1989], fuzzy set theory [Du et al., 1992], and neural networks [Tansel and McLaughlin, 1993b] to the problem of recognizing the cutting states and detecting tool breakage. Among these, neural network-based techniques have been used to detect detection of tool breakage in milling and monitoring manufacturing processes [Guillot and Ouafi, 1991].

However, most existing methods require prior knowledge about various fault conditions [Kozma et al., 1994] or tool breakage patterns [Guillot and Ouafi, 1991]. It is difficult to obtain a variety of good tool breakage patterns in an industrial environment. A robust method should detect any unacceptable (unseen) change rather than looking for specific known activity patterns. This paper proposes a new detection algorithm for tool condition monitoring in milling operations. The algorithm is based on ideas from the immune system. It is a probabilistic method that notices changes in force pattern of tools without requiring prior knowledge of what changes it is looking for. In this way it resembles the approach to novelty detection taken by ART neural architectures [Caudell and Newman, 1993]. Both neural networks and our immune system-based algorithm are biologically inspired techniques that have the capability of identifying patterns of interest. However, they use different mechanisms for recognition and learning.

In the next section, the basic immunity-based detection algorithm is described. The problem, simulated cutting force dynamics in a milling process, is discussed in section 3. In section 4, the proposed method is demonstrated for tool breakage detection by monitoring (simulated) cutting force patterns. This includes the preprocessing of sensory data and the implementation details of generating detector sets for monitoring tool conditions. Section 5 reports the results of different set of experiments and our observations in performance evaluation. Conclusions are given in section 6.

Immunity-Based Change Detection Algorithm

This detection algorithm is inspired by the information-processing properties of the natural immune system [Forrest et al., 1994]. The immune system uses learning, memory, and associative retrieval to solve pattern recognition problems. Vertebrate immune systems are capable of distinguishing virtually any foreign cell or molecule from the body's own cells which are created and circulated internally. This is known as the self-nonself discrimination problem [Percus et al., 1993]. In the immune system, T cells have receptors on their surface that can detect foreign proteins (antigens). During the generation of T cells, receptors are made by a pseudo-random genetic rearrangement process. Then they undergo a censoring process, called negative selection, in the thymus where T cells that react against self-proteins are destroyed, so only those that do not bind to self-proteins are allowed to leave the thymus. This censoring process is very important in self-nonself discrimination. Our artificial immune system [Forrest et al., 1994] is a simplification of the complex chemistry of antibody/antigen recognition in natural immune systems. The basic principle of our negative-selection algorithm is as follows:

- Define *self* as a multiset S of strings of length l over a finite alphabet, a collection that we wish to protect or monitor. For example, S may be a segmented file, or a normal pattern of activity of some system or process.
- Generate a set R of *detectors*, each of which fails to match any string in S. We use a partial matching rule, in which two strings match if and only if they are identical at least r contiguous positions, where r is a suitably chosen parameter (as described in [Forrest et al., 1994]).
- Monitor S for changes by continually matching the detectors against S. If any detector ever matches, a change (or deviation) must have occurred.

Matching Rule

We adopted a partial-matching rule based on a prespecified degree of similarity. To measure this similarity, we are currently using an r contiguous matching rule between two strings of equal length. Thus, for any two strings x and y, *match(x, y)* is true if x and y agree (match) on at least r contiguous locations (*r less than equal l*), as illustrated in figure 1.

$$X: \quad b\underline{cab}cbad$$
$$Y: \quad d\underline{cab}dcba$$

Figure 1. Illustration of Matching Rule: x and y are two strings defined over the four-letter alphabet *a, b, c, d*. X and Y match at three contiguous locations (underlined). Thus, *match(x, y)* is true for $r \leq 3$ and false for $r > 3$.

A partial-matching rule provides a detector its capability of detecting sample strings in its neighborhood according to the threshold value, r. This is demonstrated in figure 2, for the binary string. The graphs in figure 2 illustrate that the coverage of a string of defined length increases exponentially with the decrease of r. Though the maximum coverage can be achieved with $r = 1$, but the generated detectors will probably be matched with many self strings resulting in false detection. On the other hand, a perfect matching (for $r = l$) implies that symbols are identical at each location in two strings; accordingly, a very large number of detectors are needed to detect patterns in the non-self space. An optimal r value estimates a reasonable size detector set for the success of this method.

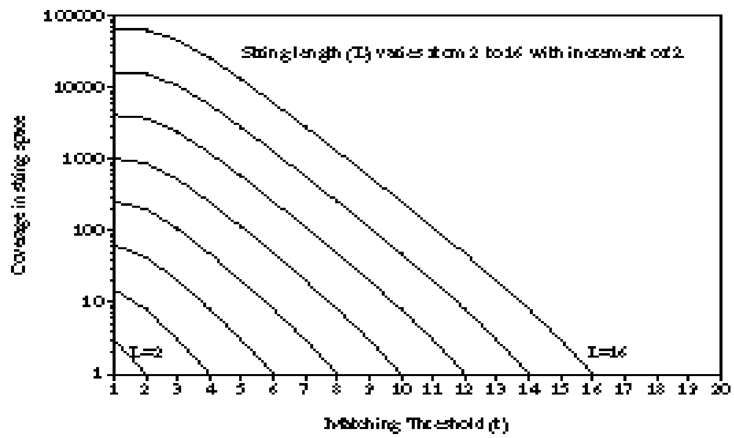


Figure 2. The figure shows (in log scale) how the number of points that can be covered by each binary string (of defined length in its string space) varies with different matching threshold.

When a non-overlapping set of detectors is generated with a suitable matching threshold, each detector can serve as a distinct novelty pattern class in the non-self space. However, in case of overlapping detectors, multiple detectors may be activated for a sample (abnormal) pattern, and need more detectors to provide sufficient enough coverage in the non-self space.

Generating Detectors

There are many possible ways to generate detectors in the non-self space. These approaches generally have different computational complexities, and their complexity is dependent on the choice of matching rule. In the original description of the negative-selection algorithm [Forrest et al., 1994], candidate detectors are generated *randomly* and then tested (censored) to see if they match any self string. If a match is found, the candidate is rejected. This process is repeated until the desired number of detectors is generated. A probabilistic analysis is used to estimate the number of detectors that are required to provide a given level of reliability. The major limitation of the random generation approach appears to be computational difficulty of generating valid detectors, which grows exponentially with the size of self. Also for many choices of l and r, and compositions of self, the random generation of strings for detectors may be prohibitive.

In this paper, we generate detector sets using an improved algorithm proposed by [Helman and Forrest, 1994] which runs in linear time with the size of self. The algorithm has two phases; first it employs a dynamic programming technique to count recurrences in order to define an enumeration of all unmatched strings (i.e. all feasible detectors). Second, a random subset of this enumeration is chosen to generate a detector set. In other words, given a collection of self strings S and matching threshold r, the first phase of the algorithm determines the total number of unmatched strings that exists for the defined self (S); then in the second phase, some of them are selected to generate detectors for monitoring self (normal patterns).

Simulation of Cutting Tool Dynamics

The dynamics of a machining process can generally be monitored when it operating in a defined environment [Altintas and Yellowley, 1989]. Usually, the methods for monitoring a milling process utilize measurements of cutting parameters correlated with tool breakage [Elbestawi et al., 1994]. These cutting parameters include temperature [Palmai, 1987], cutting force [Elbestawi et al., 1994], vibration [Moore and Reif, 1992], torque [Takata et al., 1985], acoustic emission [Liang and Dornfeld, 1989], etc. Of these parameters, cutting forces [Elbestawi et al., 1994][Tarng and Lee, 1992] [Li et al., 1992] are widely used for tool breakage detection for several reasons:

- Cutting force signals are much less dependent on the structure.
- Cutting force signals can be simulated easily and more accurately than acceleration and acoustic emission signals.
- The cutting force is a very good indicator of the vibration between the tool and workpiece because of their higher sensitivity and more rapid response to the changes in cutting state.

The cutting force variation characteristics of normal and broken tools are different. Under normal (stable) cutting conditions, the cutting force periodically varies with the tooth frequency, Ω_t that depends on the spindle speed:

$$\Omega_t = \frac{NP}{60} \tag{1}$$

where N is the spindle speed in *rpm* and P is the number of teeth on the cutter.

If the tool is broken, it can not remove the same amount of material as the other teeth. Accordingly, the number of tooth periods deviates from the stable cutting pattern depends on the number of teeth that are actively involved in the cutting zone.

Some approaches for tool breakage detection are based on the analysis of signal spectra obtained from prior FFT signal preprocessing where signal magnitude at specific frequencies increases when tool fractures occur. However, Moore and Reif [Moore and Reif, 1992] demonstrated that tool breakage can be more reliably monitored in the time-domain than in the frequency-domain.

We prepared simulated data for cutting operations using the vibratory model described in [Elbestawi et al., 1994] [Tlusty and Ismail, 1983]. This model has been used by many other investigators for tool breakage detection [Tansel and McLaughlin, 1993a][Tarng and Lee, 1992].

To generate data, the spindle was represented by a vibratory system with two degrees of freedom in the two orthogonal directions X and Y. We considered a four-tooth cutter with uniform pitch, performing an end-milling, half immersion cut in the X-direction. In this model, the instantaneous cutting force at angle φ is assumed to be proportional to the chip thickness, h. Forces acting on the tooth are the tangential force, F_t and radial force, F_r. The instantaneous tangential force F_t can be approximated by considering the tool displacement as:

$$F_t = K_c\, b\, h \tag{2}$$

subject to the condition that if $F_t \leq 0$ then $F_t = 0$.

In equation 2, K_c is the dynamic cutting force coefficient, b is the axial depth of cut, and h is the chip thickness. The instantaneous chip thickness is obtained from:

$$h = f_t \sin\varphi - z + z_{\min}$$

where f_t is the feed rate per tooth, and z is the displacement of the tool normal to the machined surface which is derived from vibratory displacements in the X and Y directions; z_{min} is the minimum undulation left behind in preceding cuts at the angle φ.

Now the displacement, z in the direction normal to the cut surface is given by:

$$z = x\sin\varphi + y\cos\varphi \tag{3}$$

The corresponding instantaneous radial component of the cutting force

$$F_r = K_r F_t \tag{4}$$

In previous studies [Elbestawi et al., 1994] [Tlusty and Ismail, 1983], the value of K_r was assumed as 0.3.

For non-helical teeth, the instantaneous cutting force in the X and Y directions can be obtained by decomposing the cutting forces F_t and F_r into the X and Y directions:

$$\begin{aligned} F_x &= F_t\cos\varphi + F_r\sin\varphi = F_t(\cos\varphi + K_r\sin\varphi) \\ F_y &= -F_t\sin\varphi + F_r\cos\varphi = F_t(-\sin\varphi + K_r\cos\varphi) \end{aligned} \tag{5}$$

In case of multi-tooth milling, the instantaneous cutting forces in the X and Y directions can be expressed as:

$$\begin{aligned} FX &= \sum_{i=1}^{P}\delta(i)F_x(\varphi_i) \\ FY &= \sum_{i=1}^{P}\delta(i)F_y(\varphi_i) \end{aligned} \tag{6}$$

and

$$\delta(i) = \begin{matrix} 1 & if\varphi_s \le \varphi_i \le \varphi_e \\ 0 & otherwise. \end{matrix} \tag{7}$$

In equation 7, φ_s, φ_e are the start and exit angle of cut, and φ_i is the cutting edge rotation angle of the i^{th} tooth.

Now, the instantaneous resultant cutting force,

$$F = (FX^2 + FY^2)^{1/2} \tag{8}$$

At every angle φ, the vibration amplitude induced by cutting forces are used where the force components of F (F_x and F_y) excite vibrations in X and Y directions which can be determined from the equations of motion for the system:

$$\begin{aligned} F_x &= m_x\ddot{x} + c_x\dot{x} + k_x x \\ F_y &= m_y\ddot{y} + c_y\dot{y} + k_y y. \end{aligned} \tag{9}$$

In equation 9, the structural parameters of two modes of vibration are m is the mass, c is the damping coefficient, and k is the stiffness. At time step t, the cutter has rotated by the angle φ_t from the reference axis Y. The F_x and F_y components of the cutting force excite vibrations in the x and y directions. The cutting force profiles were simulated using forth-order Runge-Kutta method for every time step (δt = 0.0001 sec), where displacements at step, $t+1$ are calculated from the cutting force data at step t. So the computation loop is repeated for every time step δt, then as a whole cycle per tooth period. The deflections are used to determine the uncut chip thickness for each tooth in cut (in the direction normal to the cut surface, z). Once the uncut chip thickness is determined, its value is used to determine instantaneous cutting forces.

In our experiments, one tooth is engaged in the cut at an angle φ, where the cutting angle varies from 0 to $\pi/2$ for every tooth engagement. The complete breakage of one tooth was simulated where the broken tooth did not remove any material or started to remove less material than the other teeth that gave periodic amplitude fluctuations in cutting force.

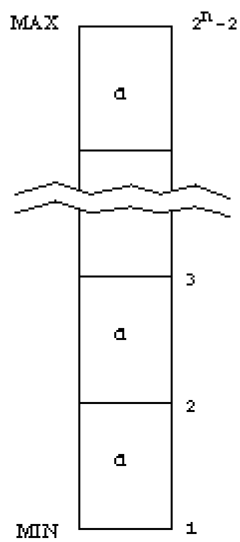
References [Elbestawi et al., 1994] [Tlusty and Ismail, 1983] can be reviewed for the detailed analysis of the vibratory model, and calculation of the cutting force and vibration. The parameter values used in our simulation are as follows:

Damping coeff., $c_x = c_y = 471.9$ kg/s;
Spring constant, $k_x = k_y = 8.1 * 10^6$ N/m;
Cutting coefficient, $K_c = 6.67 * 10^6$ N/m;
Spindle speed, $N_s = 600$ rpm;
Mass, $m_x = m_y = 10$ kg;
Feed rate/tooth, $f_t = 0.2$ mm;
Depth of cut, $b = 0.508$ mm;
Spindle diameter, $D = 40$ mm.

Tool Breakage Monitoring

We formulated the tool breakage detection problem in terms of the problem of detecting temporal changes (or abnormal patterns) in cutting force patterns resulting from the broken cutter. The patterns are encoded as strings and are monitored for whether or not the current strings are different (matched using negative selection), where a change (or match) implies a shift in the normal behavior in cutting force patterns.

Data Preprocessing:



A Coding Scheme

Figure 3: Illustration of a mapping technique for encoding close analog values into a discrete form. For binary encoding with n bits/data, the number of intervals and the size of each interval (d) are shown here.

We first preprocess sensory data into a form suitable for our detection algorithm. Preprocessing can be viewed as constructing an alternative representation in an attempt to capture the regularities of the data while preserving the information content. Further, any change that exceeds allowable variation in the data pattern should ideally be reflected in the representation space. This can be a problem when perhaps very small changes in real-valued data need to be monitored. To handle this, we use an approach that maps close real-valued data into a discrete form: an analog value is normalized with respect to a defined range and discretized into bins (or intervals). Each datum is assigned the integer corresponding to the bin in which it falls.

The integer is then encoded using binary representation. However, if an observed value falls outside the specified range, it is mapped to all 0's or all 1's depending on which side of the range it crossed. The number of bits used in the discretization thus determines the size of the bins. If each datum is encoded by n bits (which may be chosen according to the desired precision), then there are $2^n - 2$ different bins between the maximum *(MAX)* and minimum *(MIN)* ranges of data (see figure 3).

Implementation Details

In our implementation, raw sensory data are sampled from a moving time window and mapped to binary form. Each window, therefore, is the concatenation of a fixed number (called *Win_size*) of data points. We collect the bit strings from a succession of windows, sliding along the time series in discrete steps (*Win_shift*) for the normal data set. As long as the time series data pattern maintains similar behavior, these

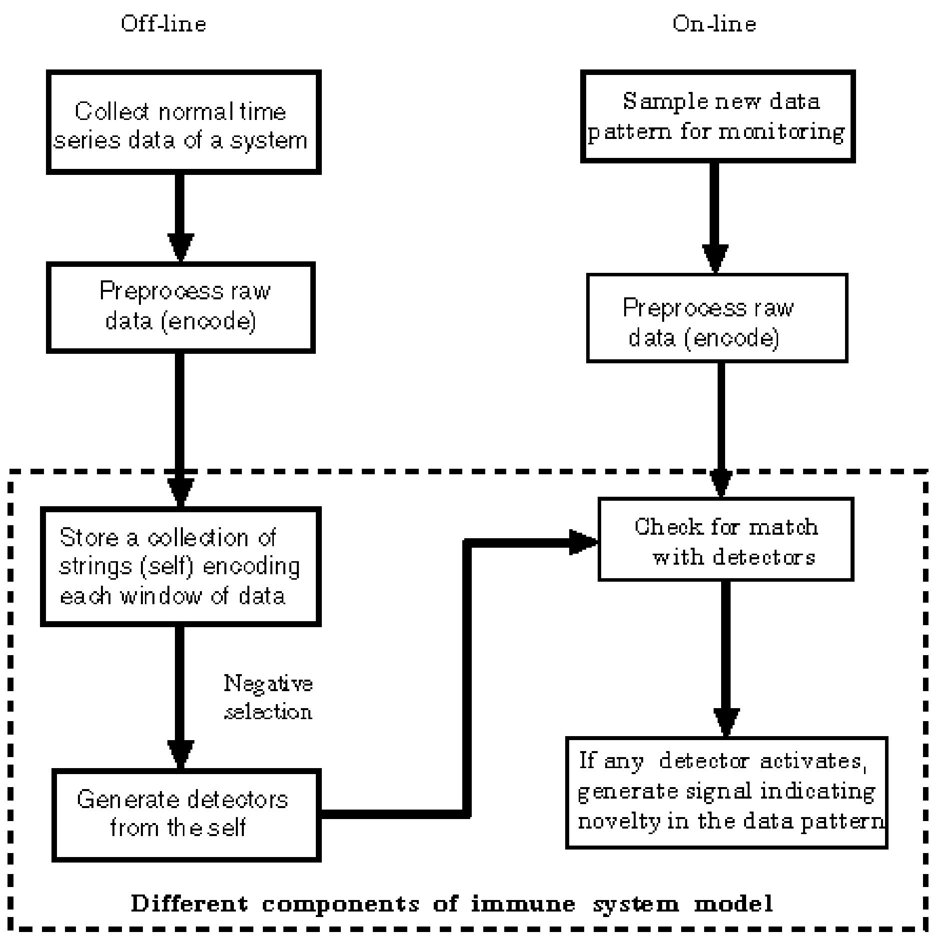


Figure 4. Schematic diagram showing the processing stages of the immune system based fault detection.

collected strings are sufficient to define normal behavior of the system. This collection of strings for windows is our self set (*S*). We then generate strings that do not match any of the strings in *S* to be members of the detector set. The generation of detectors in this detection algorithm is usually performed off-line, as in the case of neural networks (supervised) training for fault detection [Guillot and Ouafi, 1991] or developing rule-based expert-systems for detecting faults/anomalies etc. [Frank, 1990]. Overall, our approach can be summarized as follows (see figure 4):

1. Collect time series (sensor) data that sufficiently exhibit the normal behavior of a system (these may be raw data at each time step, or average values over a longer time interval).
2. Examine the data series to determine the range of variation (*MAX, MIN* values) of data and choose the data encoding parameter (n) according to the desired precision.
3. Encode each value in binary form using the above coding scheme.
4. Consider a suitable window size that can capture the semantics in data pattern.
5. Slide the window along the time series and store the encoded string for each window as *self* for processing by the negative-selection algorithm.
6. Generate a set of detectors that do not match any of the self strings according to the partial matching rule with suitably chosen r. It is desirable that the detectors are spread enough to cover the unmatched string (non-self) space. Also an estimate for the size of the detector set is needed to ensure a certain level of reliability in detecting changes [Forrest et al. 1994].
7. Once a unique set of detectors is generated from the normal database, it can probabilistically detect any change (or abnormality) in patterns of monitoring sensory data.
8. When monitoring the system, we used the same preprocessing parameters as in step 3 and 4, to encode new data patterns (moving window). If a detector is ever activated (matched with current pattern), a change in behavior pattern is known to have occurred and an alarm signal is generated regarding the abnormality. We use the same matching rule (for monitoring the system) as was used in generating detectors.

Also encoding parameters that affect preprocessing are:

BITS_PER_DATA (n) - this will dictate the degree of numerical precision with which real numbers are represented in binary form. For example, 5-bit data encoding gives 30 intervals into which the range *[MIN, MAX]* of data is divided.

WINDOW_SIZE (w) - the number of samples encoded in a single pattern (each string in self).

WINDOW_SHIFT - the number of samples by which one pattern is shifted from the previous one in a moving window. For example, if WIN_SHIFT = 1 with a window size w, the patterns will be $\{x_1, x_2, ..., x_w\}$, $\{x_2, x_3, ..., x_{(w+1)}\}$ .... etc.

Experimental Results

We simulated several test cases of the milling cutter dynamics to carry out a set of experiments with the proposed detection algorithm. The purpose is to detect tool breakage in different cutting environment.

Figure 5 shows typical cutting force patterns with and without tool breakage in a simulated milling operation. In this simulation, the tool was in normal cutting operation for 1500 time steps and then one tooth was broken, causing changes in cutting force signals at the corresponding tooth periods. In our experiments, we used the first 1000 data points as the self set, *S* for generating detectors and the rest of the data series are used for testing. Results of the experiments are shown in table 1 and in figure 6. Table 1 shows the various parameters used for preprocessing data and for generating detectors. We tried several different parameter values and found the reported values most suitable. We generated the diverse set of detectors in such a way that they do not match each other by r-contiguous bit rule. In these experiments, we set n = 6 for binary encoding of data and two different window sizes are considered.

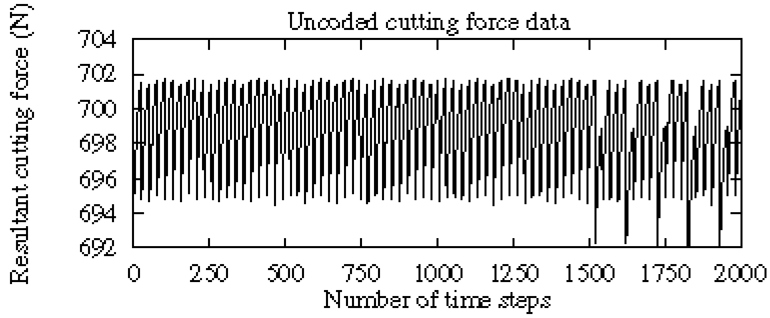


Figure 5. Simulated cutting force signals of normal and with tool breakage in a milling operation. Here one tooth of the cutter is broken after 1500 time steps.

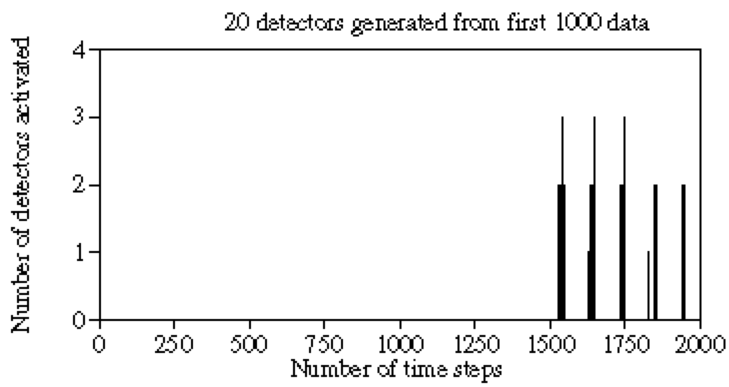


Figure 6: The height of vertical lines in the graph corresponds to the number of detectors activated when novel patterns are found.

Encoding parameters	Matching Threshold (r)	Number of Detectors (R)	Breakage detection	
			Mean (Std. Dev.)	Detection Rate
Win_size=5	10	50	14.30(2.32)	59.58%
Win_shift=5	9	40	17.57(2.25)	74.32%
l =30, S=200	8	30	22.16(2.57)	91.64%
Win_size=7	12	40	10.36(3.36)	62.78%
Win_shift=7	10	30	20.38(5.57)	75.56%
l =42, S=142	9	20	30.75(7.91)	93.28%

Table 1: Tool breakage detection results, averaged over 50 runs. Column 4 shows the mean number of detections (number of times detectors activated). The standard deviations are shown in parentheses. The detection rate is shown in column 5. This is the ratio of the average detection to the number of actual novel patterns in data.

Detection results (columns 4 and 5) show that the mean number of times detectors activated and the average detection rate in each case. In all the test runs, the generated detectors could detect the tooth periods in which the changes in the force pattern occurred. Figure 6 shows a typical run and the number of activated detectors (novel patterns encountered) at different time steps. In this example, a maximum of three detectors is activated (out of 20) when there are significant changes. Note that the detectors remain inactive during normal operation period, in particular, between 1000 and 1500 time steps where the data exhibit a normal pattern, thus avoiding false positives. Also all the broken tooth periods could easily be detected.

Further experiments were conducted with various cutting parameters to simulate cutting forces for normal and broken tool condition, these are summarized in Table 2. In each case, first 1500 data were considered as the measurement of normal cutting and the rest were for the broken the tool. In all these experiments, encoding parameter *n* was set to 5. Two different window sizes were considered with different parameter setting. Experiments were repeated (10 times) for each cutting condition, where a small set of detectors were generated from 1000 initial data and used for monitoring the rest. Results of the experiments are shown in Tables 3. These experiments indicate that our algorithm can easily detect tool breakage in all test cases.

Experiment Number	Axial Depth Cut (mm)	Feed Rate (m/min)	Spindle Speed(rpm)	Spindle Diameter(mm)
1	1.34	90.6	800	50
2	1.016	125.4	500	40
3	1.524	50.8	700	40

Table 2: Use of various cutting parameters for generating cutting force signals.

Encoding Parameters	Experiment Number	Matching Threshold(r)	Total Number Of Detectors Generated	Percentage of	
				Detectors Activated	Broken Periods Detected
Win_size = 6	1	8	30	75%	98%
Win_shift =6	2	8	40	72%	100%
l=30, N_s=166	3	9	50	63%	96%
Win_size = 8	1	9	40	78%	99%
Win_shift =8	2	10	50	73%	98%
l=40, N_s=125	3	11	70	67%	95%

Table 3: The table shows results of tool breakage detection under different cutting condition. Columns 5 and 6 show the number (in p.c.) of detectors that were activated when novel patterns encountered in tooth periods that correspond to the broken tooth and the detection rates. It is to be noted that the detection rates were high (varied between 95%-100%) while monitoring broken tooth periods.

The observed results agreed with our theoretical predication that the performance of the algorithm varies with the choice of the matching threshold (r). With larger r, the generated detectors become sensitive to any particular novelty in data patterns, so more detectors are necessary to achieve a desired level of overall reliability. On the other hand, if r is too small, it may not be possible to generate a detector set from the available self of reasonable size, since there may not exist any unmatched strings (non-self) at that value of r. This suggests that the value of r can be used to tune the reliability of detection against the risk of false positives.

Conclusions

In this paper, we have proposed a method for tool breakage detection based on principles inspired by the natural immune system. The objective of this work is to develop an efficient detection algorithm that can be used to alert an operator to any changes in steady-state characteristics of milling cutter dynamics. The results demonstrated that the proposed algorithm could successfully detect the tooth breakage from dynamic variation of the cutting force signals. It is to be noted that our approach relies on a large enough samples of normal sensory data to generate a diverse set of detectors that probabilistically notice any deviation from the normal operation. Because it does not look for any particular (or known) fault, rather indicate that these patterns are novel with respect to the normal behavior pattern, this algorithm could be incorporated into existing diagnostic tools for further classification. The detection system can quickly be updated by generating a new set of detectors as the normal milling operation shifts due to modifications of tool-workpiece geometry, change in the cutting condition, etc. [Forrest et al., 1994] showed that a small set of detectors can have a very high probability of noticing changes to the original data set.

In most monitoring systems, the detection of a spurious change in sensor measurements is not as important as the gradual change in the pattern over a period of time, so our probabilistic detection algorithm appears to be a promising alternative approach to such problems. Also it may be needed to choose a fault-detection threshold in order to allow instantaneous variations or spikes in new patterns from the established normal patterns while monitoring real sensor data.

There are a number of parameters that are tunable in both the preprocessing and the detector generation stage. In the preprocessing stage, the desired precision can be achieved by grouping similar analog data in the same bin, and the window size may be suitably chosen to capture the regularities of the data patterns. Note that the system can be monitored using different time scales simultaneously. Also, instead of directly encoding the time series data, it may be necessary to transform data (e.g. by Fourier transform) depending on the properties of sensor data. It is also possible to combine several sensor data (i.e. sensor fusion) in order to improve the reliability of the monitoring system [Dornfeld, 1990]. Particularly, when a single sensor cannot provide a good correlation with all the anomalies that need to be detected. Also the decisions based on multiple sensors will provide more information simultaneously, the quality will more likely be better than the decisions based on a single sensor. One simple data fusion technique is a weighted addition of the sensor signals. A desired level of detection reliability can be achieved by changing the window size, matching threshold, and the number of detectors. The probability of a match at r contiguous positions and the impact of different choices of r on the overall computational behavior of the algorithm are reported in [Forrest et al. 1994]. Theoretical analysis and empirical experiments suggest that the algorithm is highly sensitive to the value of r. We are currently investigating other matching rules and generation algorithms as a future research[1].

We have tested the feasibility of this detection algorithm on a number of data sets, including the Mackey Glass series [Caudell and Newman, 1993], and some real sensor data. These experiments suggest that this detection algorithm may be useful for many other similar problems. They include fault detection, anomaly detection, machine monitoring, signature verification, noise detection, patient's condition monitoring and so forth. It should be pointed out that the idea of using immune system principles in fault detection was also studied by others [Ishida and Mizessyn, 1992][Ishida, 1993], however, they have chosen a different set of

[1] A short version of the results presented in this paper was reported in [Dasgupta and Forrest, 1996].

principles to emulate the process fault diagnosis. The remarkable detection abilities of biological immune systems suggest negative-selection algorithms such as ours be well worth exploring in industrial applications.

References:

[Altintas and Yellowley, 1989] Altintas, Y. and Yellowley, I. (1989). In-Process Detection of Tool Failure in Milling using Cutting Force Models. In *Journal of Engineering for Industry*, 111:149--157.

[Caudell and Newman, 1993] Caudell, T. P. and Newman, D. S. (1993). An Adaptive Resonance Architecture to Define Normality and Detect Novelties in Time Series and Databases. In *IEEE World Congress on Neural Networks*, pages IV166--176, Portland, Oregon.

[Chryssolouris and Guillot, 1990] Chryssolouris, G. and Guillot, M. (1990). A Comparison of Statistical and AI approaches to the Selection of Process Parameters in Intelligent Machining. In *Transactions of the ASME; Journal of Engineering for Industry*, 112:122--130.

[Dagli and Poshyanonda, 1994] Dagli, C. H. and Poshyanonda, P. (1994). *Artificial Neural Networks for Intelligent Manufacturing*, chapter-3, page 57, Chapman & Hall.

[Dasgupta and Forrest, 1996] Dasgupta, D. and Forrest, S. (1996). Novelty detection in time series data using ideas from immunology. In *ISCA 5th International Conference on Intelligent Systems*, Reno, Nevada.

[Dornfeld, 1990] Dornfeld, D. A. (1990). Neural Network Sensor Fusion for Tool Condition Monitoring. In *Annals of the CIRP*, 24:101--105.

[Du et al., 1992] Du, R. X., Elbestawi, M. A., and Li, S. (1992). Tool condition monitoring in turning using fuzzy set theory. In *International Journal of Machine Tools & Manufacturing*,32(6):781--796.

[Elbestawi et al., 1994] Elbestawi, M. A., Ismail, F., Du, R., and Ullagaddi, B. C. (1994). Modelling machining dynamics including damping in the tool-workpiece interface. In *Journal of Engineering for Industry*, 116:435--439.

[Elbestawi et al., 1991] Elbestawi, M. A., Papazafiriou, T. A., and Du, R. X. (1991). In-process monitoring of tool wear in milling using cutting force signature. In International Journal of Machine Tools and Manufacturing, 31(1):55--73.

[Forrest et al., 1994] Forrest, S., Perelson, A. S., Allen, L., and Cherukuri, R. (1994). Self-Nonself Discrimination in a Computer. *In Proceedings of IEEE Symposium on Research in Security and Privacy*, pages 202--212, Oakland, CA.

[Frank, 1990] Frank, P. M. (1990). Fault Diagnosis in Dynamic Systems using Analytical and Knowledge-based Redundancy - A survey and some new results. In *Automatica*, 26(3):459--474.

[Guillot and Ouafi, 1991] Guillot, M. and Ouafi, A. E. (1991). On-line Identification of Tool Breakage in Metal Cutting Processes by use of Neural Networks. In *Intelligent Engineering Systems Through Artificial Neural Networks*, volume-I, pages 701--709, ASME Press, New York.

[Hajela et al. 1997] Hajela, P., Yoo, J. and Lee, J. (1997). GA Based Simulation of Immune Networks - Applications in Structural Optimization. In *Journal of Engineering Optimization*.

[Hart et al. 1998] Hart, E., Ross, P. and Nelson, J. (1998). Producing robust schedules via an Artificial Immune system. *In the proceedings of the IEEE International Conference on Evolutionary Computation*.

[Helman and Forrest, 1994] Helman, P. and Forrest, S. (1994). An Efficient Algorithm for Generating Random Antibody Strings. Technical Report Technical Report No. CS94-7, Department of Computer Science, University of New Mexico.

[Ishida, 1993] Ishida, Y. (1993). An Immune Network Model and its Applications to Process Diagnosis. In *Systems and Computers in Japan*, 24(6):38--45.

[Ishida and Mizessyn, 1992] Ishida, Y. and Mizessyn, F. (1992). Learning Algorithms on an Immune Network Model: Application to Sensor Diagnosis.
In *Proceedings of International Joint Conference on Neural Networks*, volume~I, pages 33--38, China.

[Kozma et al., 1994] Kozma, R., Kitamura, M., Sakuma, M., and Yokoyama, Y. (1994). Anomaly {D}etection by neural network models and statistical time series analysis. In *Proceedings of IEEE International Conference on Neural Networks*, Orlando, Florida.

[Li and Wu, 1989] Li, C. J. and Wu, S. M. (1989). On-line Detection of Localized defects in bearings by pattern recognition analysis. In *Transactions of the ASME; Journal of Engineering for Industry*, 1121:331--336.

[Li et~al., 1992] Li, G. S., Lau, W. S., and Zhang, Y.Z. (1992). In-Process Drill wear and breakage monitoring for a machining centre based on cutting force parameters. In *International Journal of Machine Tools and Manufacturing*, 32(6):855--867.

[Liang and Dornfeld, 1989] Liang, S. Y. and Dornfeld, D. A. (1989). Tool wear detection using time series analysis of acoustic emission. *In Journal of Engineering for Industry*, 111:199--205.

[Moore and Reif, 1992] Moore, T. and Reif, Z. (1992). Detection of Tool Breakage using Vibration Data. In *Proceedings of North American Manufacturing Research Conference (13th NAMRC); SME Transactions on Manufacture Engeering*, pages 45--50.

[Palmai, 1987] Palmai, Z. (1987). Cutting Temperature in Intermittent Cutting. In *International Journal of Machine Tools & Manufacturing*, 27(2):261--274.

[Percus et al., 1993] Percus, J. K., Percus, O., and Person, A. S. (1993). Predicting the size of the antibody combining region from consideration of efficient self/non-self discrimination. In *Proceedings of the National Academy of Science*, 60:1691--1695.

[Rangwala and Dornfeld, 1990] Rangwala, S. and Dornfeld, D. (1990). Sensor integration using neural networks for intelligent tool condition monitoring. In *Journal of Engineering for Industry*, 112:219--228.

[Takata et al., 1985] Takata, S., Ogawa, M., Bertok, P., Ootsuka, J., Matushima, K., and Sata, T. (1985). Real-Time Monitoring System of Tool Breakage using Kalman Filtering. In *Robotics & Computer-Integrated Manufacturing*,2(1):33--40.

[Tansel and McLaughlin, 1993a] Tansel, I. N. and McLaughlin, C. (1993a). Detection of tool breakage in milling operations-I. The time series analysis approach. In *International Journal of Machine Tools & Manufacturing*, 33(4):531--544.

[Tansel and McLaughlin, 1993b] Tansel, I. N. and McLaughlin, C. (1993b). Detection of tool breakage in milling operations-II. The neural network approach. In *International Journal of Machine Tools & Manufacturing*, 33(4):545--558.

[Tarng and Lee, 1992] Tarng, Y. S. and Lee, B. Y. (1992). Use of model-based cutting simulation system for tool breakage monitoring in milling. In *International Journal of Machine Tools & Manufacturing*,32(5):641--649.

[Tlusty and Ismail, 1983] Tlusty, J. and Ismail, F. (1983). Special aspects of chatter in milling. In *Trans. ASME; Journal of Vibration, Acoustics, Stress, and Reliability in Design*, 105:24--31.