# Control Methods to Complete Jobs at Desired Times

Wei Weng and Shigeru Fujimura

***Abstract*—This paper proposes control methods in a flexible job shop in the aim of completing the jobs at predetermined times. The predetermined times can be the times when the jobs can start undergoing processing by the next production process on the production line or the times when the jobs should be delivered to the customer or the next party on the supply chain. By completing the jobs at such times, the flow of jobs or goods between production processes or supply-chain parties can be smoothed and the intermediate inventory of finished jobs or goods reduced. The proposed control methods are performed online, featuring high speed, requirement of little beforehand information, and the ability to adapt to disturbances. The methods include a dispatching rule which sorts the jobs based on the forecast waiting time of each job and a decision making process which gives priority to the jobs which are waiting for the processing of their final operations. Computer simulations are conducted using the data of a real factory and the results show that the methods are higher in performance than the existing methods for the objective.**

***Index Terms*—Flexible job shop, online production control, industrial case study.**

## I. Introduction

In the manufacturing industry, the making of products may require several relatively independent production processes, which in some cases are performed by several production shops respectively. Most existing studies in the production control area focused on the optimization of one shop only [1]-[2]. Such an optimization may not lead to the optimization of the whole production line if there are successor shops. An example is that the achievement of zero inventories in one shop may not contribute to the reduction of inventories on the production line. The same holds true when considering the optimization of the flow of goods along the parties on a supply chain.

To improve the performance of the whole production line, the objective of each shop should be set according to the objective of the production line. For example, the speed at which the goods are made should be set according to the speed at which the goods can be sold, and the times when the jobs are completed in a shop should be determined by the times when the jobs are wanted by the successor shop. This is similar to a pull process.

By investigating the literature on the methods developed for the pull process, it is found that though the concept of pull production was proposed long ago [3] and is proven to be beneficial to production systems [4]-[5], the studies in the area mainly worked on the optimization of a shop in which the jobs go through all the machines in a common sequence [6]-[8]. In such a case, the market demand can be used to pull the jobs out of the last machine in the system, and the demand of each machine be used to pull the jobs out of its predecessor machine. The process continues until the raw material is pulled into the production shop.

In real factories, however, though the production shops are usually linked serially on the production line, the machines inside one production shop are not always linked serially. For example, in the extensively used job shop environment, each job has a unique route to go through the system, and some jobs may skip or revisit some of the machines. In such a complex shop system, it is impossible to implement the pull process between the machines. Consequently, the existing methods used in the pull production systems cannot control the jobs in a complex shop to be completed at the times according to the demand.

Based on the investigation of a number of factories in Japan and Taiwan, it is found that the problem widely exists that it is difficult to control the processing of jobs in a complex shop so as to meet the need of the others shops on the production line.

To provide a solution to this problem, this study proposes some methods which can relatively easily be implemented in a complex shop to achieve the objective. We design the methods based on a flexible job shop (FJS) used in a real factory. The FJS is one of the complex shops that are widely used in the industries [9]-[11] and existing studies on FJS did not focus on the mentioned problem.

The methods are performed online and require little information about the production system environment and process. Such a design is suitable for complex shop systems for which offline scheduling may take a long computation time and is unable to respond to disturbances. The methods sort the jobs in processing based on the forecast waiting time in remained operations of each job and select the next job to process with priority given to the jobs that are waiting for the processing of their final operations. Computer simulations are carried out by using the data of a factory and the results show that the methods are much higher in performance than the existing online control rules.

The remainder of this paper is organized as follows: section II gives the detailed problem description; section III proposes the online control methods; section IV presents the computer simulations with analysis and discussions; and section V summarizes the findings and suggests some applications and future topics.

## II. Problem Description

Notation

$C_j$ completion time of job $j$

Manuscript received July 1, 2012; revised August 28, 2012.

Wei Weng and Shigeru Fujimura are with the Graduate School of Information, Production and Systems, Waseda University, Fukuoka 808-0135, Japan (e-mail: wengwei@toki.waseda.jp).

DOI: 10.7763/IJIEE.2012.V2.239

Dj due date of job j
j index of job, j = 1, 2, ..., J
m index of machine, m = 1, 2, ... |Ms|
o index of operation, o = 1, 2, ..., O
s index of station, s = 1, 2, ..., S

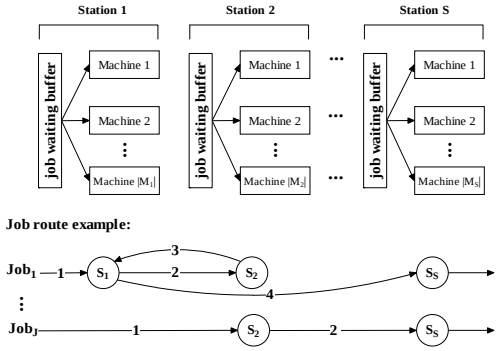



Fig. 1. Environment and process of the flexible job shop problem.

This study focuses on an FJS along a production line. The shop is composed of $S$ stations, with $|M_s|$ identical machines and a job waiting buffer at station $s$. Each job has a number of operations ($o$ = 1, 2, ..., $O$), each of which must be processed in a specified station. In a station, a job can be processed by any one of the $|M_s|$ machines and the processing time is the same. The machines at some stations are allowed to process multiple jobs simultaneously as long as the total volume of the jobs does not exceed an upper limit whereas at the other stations one job at a time. The transportation time between stations is neglected, and a job, after being completed, will go to a successor shop for further processing. Fig. 1 shows the environment and process of the FJS problem.

The time job $j$ is wanted by the successor shop is set as the job's due date $D_j$. Let $C_j$ denote the job's completion time, then if job $j$ is completed earlier than $D_j$, it has an earliness $E_j = D_j - C_j$; otherwise, a tardiness $T_j = C_j - D_j$. The objective is to minimize the average earliness and tardiness of all the jobs, as expressed by

$$\min \sum_{j=1}^{J} \max(E_j, T_j) / J \ . \qquad (1)$$

The following are the assumptions that are made in this study.

Assumptions

1) The processing of a job is non-preemptive.
2) Machines do not fail down.
3) The waiting buffer in each station is unlimited.

## III. Proposed Control Methods

In order to control the jobs towards on-time completion, we proposed two online control methods. One method orders the jobs in the waiting buffer in each station, and the other method helps a machine select from the waiting jobs the next job to process. The notation used in the methods is given below in the alphabetical order.

Notation

$C$ current operation of a job
$DM_j$ destination machine of job $j$
$J_m$ jobs that are being processed by machine $m$
$Q_s$ set of the jobs that are waiting at station $s$
$R_j$ forecast remained time of job $j$
$R_{jo}$ remained processing time of operation $o$ of job $j$
$s(j,o)$ station where operation $o$ of job $j$ is processed
$T$ time at present
$T_{jo}$ processing time of operation $o$ of job $j$
$V_m^A$ available space in machine $m$
$V_j$ volume of job $j$
$W_{jm}$ waiting time of job $j$ at machine $m$
$W_{jo}$ waiting time of job $j$ at the station where operation $o$ is processed

The method which orders the waiting jobs is designed as follows. We calculate the waiting time of job $j$ in the station where its current operation $C$ is processed, $W_{jC}$, and forecast the waiting time for each of the job's remained operations to be the same as $W_{jC}$. This is because in a well balanced system, the waiting time at each station is generally similar.

Method to calculate the waiting time of a job if the machine can process multiple jobs simultaneously

**Input**: $Q_{s(j,C)}$, $M_{s(j,C)}$
**Output**: $W_{jC}$

Set $W_{jC} = 0$, $Q_{s(j,C)} = Q_{s(j,C)} + \{j\}$;
**for** $i \leftarrow 1$ **to** $|Q_{s(j,C)}|$ **do**
  **for** $m \leftarrow 1$ **to** $|M_{s(j,C)}|$ **do**
    Sort the jobs $\in J_m$ into the increasing order of $R_{kC}$;
    Set $W_{im} = 0$, $k = 1$;
    **while** $V_m^A < V_i$ **do**
      $W_{im} = R_{kC}$;
      $V_m^A = V_m^A + V_k$;
      $k = k + 1$;
    **end**
  **end**
  $W_{iC} = \min_{m \in M_{s(j,C)}} W_{im}$;
  $DM_i = \arg\min_{m \in M_{s(j,C)}} W_{im}$;
  $W_{jC} = W_{jC} + W_{iC}$;
  $J_{DM_i} = J_{DM_i} + \{i\}$;
  $V_{DM_i}^A = V_{DM_i}^A - V_i$;
  $R_{iC} = R_{iC} - T_{iC}$;
  **for** $m \leftarrow 1$ **to** $|M_{s(j,C)}|$ **do**
    **for** $k \leftarrow 1$ **to** $|J_m|$ **do**
      $R_{kC} = R_{kC} - W_{iC}$;
      **if** $R_{kC} \leq 0$ **then** $J_m = J_m - \{k\}$;
    **end**
  **end**
**end**
**return** $W_{jC}$;

As a result, the remained time of job $j$ can be estimated to be

$$R_j = \sum_{o=C}^{O} (W_{jC} + T_{jo}) \ . \qquad (2)$$

In the equation, $W_{jC}$ is calculated in real time. In the case that each machine at the station is only allowed to process one job at a time, $W_{jC}$ is approximated to be

$$W_{jC} = \Big( \sum_{i=1}^{|Q_{s(j,C)}|} T_{iC} \Big) / |M_{s(j,C)}| \ , \qquad (3)$$

In which $i$ is the index of job which waits in the station $s(j,$

*C*) at the time when job *j* arrives (job *j* excluded).

In the case that each machine in the station is allowed to process multiple jobs at a time, it is difficult to express $W_{jC}$ by using a simple mathematical equation. In such a case, we calculate $W_{jC}$ by using the following process, which sums up the time required before job *j* can start undergoing processing. In the pseudo code, *i* is the index of the jobs in $Q_{s(j,C)}$ and *k* is the index of the jobs in machine *m*.

In the code, lines 3-9 calculate the time that job *i* should wait before starting undergoing processing by machine *m*. Lines 10-11 select the machine where the waiting time is the shortest to be *i*'s destination. Line 12 adds the waiting time of *i* into the waiting time of job *j*. Lines 13-19 update the jobs and the machines in the station into the status after time $W_{iC}$ and remove the jobs that will be finished by the time. If there remain jobs in the queue, the process will be repeated to calculate the waiting time of the next *i*. When the process comes to the last job in the queue, the total waiting time of job *j* becomes known.

We then use the obtained remained time $R_j$ to calculate $L_j$, the slack time per remained operation of job *j*.

$$L_j = (D_j - T - R_j)/(O - C + 1) \qquad (4)$$

In the equation, ($D_j$ - $T$ - $R_j$) is the slack time between the due date and the forecast remained time of job *j*, and ($O$ - $C$ +1) is the number of the remained operations.

A positive $L_j$ represents the time flexibility that job *j* has for each remained operation whereas a negative $L_j$ is the time that job *j* should try to reduce from its $W_{jo}$ at each remained operation. We sort the jobs in the increasing order of $L_j$ in each waiting buffer.

Another control method is performed when a machine becomes idle and will select from the waiting jobs the next job to process. At the time, the machine will give priority to the jobs that are waiting for the processing of their final operations.

In detail, besides sorting the waiting jobs in the increasing $L_j$ order, we further place the final-operation jobs, if any, at the front of each waiting queue in the $L_j$ order. The idle machine will first check if any of the final-operation jobs will become tardy if it starts to process a non-final-operation job. If so, the machine will start processing the first final-operation job. Otherwise, it will start processing the first non-final-operation job.

## IV. Computer Simulations and Analysis

The proposed methods are evaluated by using the data of a factory in Japan. The methods are simulated for one of the shops in the factory, which is an FJS composed of 21 stations.

The jobs in the shop should be completed and delivered to the successor shop within 16 hours, during which time there are 5 machines available in one of the stations and one machine available in each of the other stations. The 5 machines can only process one job at a time whereas the other machines multiple jobs simultaneously.

At present, the jobs in the shop are processed in an experience- based order. Usually, most jobs are completed far earlier than the time they are wanted by the successor shop and hence have to wait for a long time before being able to start undergoing processing by the successor shop. In spite of this, a small number of jobs are completed later than the time when they are required and hence cause the successor shop to wait for them.

We apply the proposed methods to control the jobs in the shop. For evaluation, we observe the performance measures that are given in Table I. In the table, *No.* is the index of the data set, i.e. totally there are 12 sets of data. $\overline{ET}$ is the average objective function value of the 12 data sets and $\sigma_{et}$ is the standard deviation of $\overline{ET}$ . $J_o$, $J_e$ and $J_t$ are the numbers of on-time, early and tardy jobs, respectively. $\overline{E}_e$ and $\overline{T}_t$ are the average earliness of the early jobs and the average tardiness of the tardy jobs, respectively. $\sigma_e$ and $\sigma_t$ are the standard deviations of $\overline{E}_e$ and $\overline{T}_t$, respectively.

For comparison, since the experience-based job processing used on the shop floor is difficult to simulate, we select three applicable online control rules - EDD, MDD and MST. This is because among the commonly used rules including FIFO, EDD, MDD, SPT, LPT, and MST, EDD, MDD and MST are believed to deliver comparatively good performance for the just-in-time objective [12]. In pilot tests, the performance of the rules EDD and MDD is similar to that on the real shop floor.

The simulation results are given in Table I, in which PCM stands for the proposed control methods. We sort the sets of data according to the increasing order of the average time flexibility between the total processing time and the desired completion time of a job.

It can be observed that $\overline{ET}$ increases under the existing control rules whereas decreases under the proposed methods. This is because the increase in time flexibility causes the average earliness to rise under the existing methods whereas under the proposed methods, the early jobs are controlled towards being on time by adjusting the start times of their final operations.

In the case (*No.*1) in which the performance gap between the existing rules and the proposed methods is the narrowest, the $\overline{ET}$ of the proposed methods is 62.3% lower than that of the best performing existing rule MST, showing a significant superiority. Compared with the average length of the total processing time of a job, 31.8 is about 7.5% in length.
In the other cases, the performance gap between the existing and proposed methods is wider. The number of on-time jobs under the proposed methods is significantly larger than those under the existing methods. Meanwhile, the number of tardy jobs is smaller than those under the existing methods.

On average, the $\overline{ET}$ , $\overline{E}_e$ and $\overline{T}_t$ of the proposed methods are 90.5%, 92.2%, and 20.0% lower than those of the best performing existing rule MST. This reveals the effectiveness of the proposed methods, which order the jobs in a more reasonable sequence in the processing and gives priority to the final-operation jobs when selecting the next job to process. In addition, the standard deviations of the proposed methods are also the smallest among all the methods, proving the stability in performance.

TABLE I: PERFORMANCE COMPARISON BETWEEN EXISTING ONLINE CONTROL RULES AND THE PROPOSED METHODS

| | No. | ET | $\sigma_{et}$ | $J_o$ | $J_e$ | $J_t$ | $E_e$ | $\sigma_e$ | $T_t$ | $\sigma_t$ |
|---|---|---|---|---|---|---|---|---|---|---|
| EDD | 1 | 90.7 | 70.70 | 0 | 19 | 12 | 67.6 | 37.05 | 127.3 | 94.90 |
| | 2 | 95.8 | 63.98 | 0 | 20 | 11 | 83.7 | 40.01 | 117.9 | 91.69 |
| | 3 | 103.2 | 59.13 | 0 | 22 | 9 | 95.4 | 46.68 | 122.3 | 82.53 |
| | 4 | 111.6 | 58.12 | 0 | 22 | 9 | 115.4 | 46.68 | 102.3 | 82.53 |
| | 5 | 120.7 | 61.35 | 0 | 24 | 7 | 124.5 | 57.75 | 107.4 | 75.96 |
| | 6 | 131.7 | 65.44 | 0 | 25 | 6 | 138.8 | 63.40 | 102.2 | 71.42 |
| | 7 | 145.2 | 68.17 | 0 | 26 | 5 | 153.4 | 67.90 | 102.4 | 57.49 |
| | 8 | 158.7 | 73.77 | 0 | 26 | 5 | 173.4 | 67.90 | 82.4 | 57.49 |
| | 9 | 172.3 | 81.75 | 0 | 26 | 5 | 193.4 | 67.90 | 62.4 | 57.49 |
| | 10 | 185.8 | 91.50 | 0 | 26 | 5 | 213.4 | 67.90 | 42.4 | 57.49 |
| | 11 | 201.5 | 98.25 | 0 | 28 | 3 | 217.9 | 86.73 | 48.0 | 64.37 |
| | 12 | 218.7 | 102.48 | 0 | 30 | 1 | 222.6 | 101.87 | 102.0 | - |
| Mean | | 144.7 | 74.55 | 0.0 | 24.5 | 6.5 | 150.0 | 62.65 | 93.3 | 66.11 |
| MDD | 1 | 90.5 | 70.84 | 0 | 19 | 12 | 67.6 | 37.05 | 126.8 | 95.36 |
| | 2 | 95.7 | 64.20 | 0 | 20 | 11 | 83.7 | 40.01 | 117.5 | 92.27 |
| | 3 | 103.4 | 58.87 | 0 | 22 | 9 | 95.6 | 46.24 | 122.3 | 82.53 |
| | 4 | 111.6 | 58.12 | 0 | 22 | 9 | 115.4 | 46.68 | 102.3 | 82.53 |
| | 5 | 120.7 | 61.35 | 0 | 24 | 7 | 124.5 | 57.75 | 107.4 | 75.96 |
| | 6 | 131.7 | 65.44 | 0 | 25 | 6 | 138.8 | 63.40 | 102.2 | 71.42 |
| | 7 | 145.2 | 68.17 | 0 | 26 | 5 | 153.4 | 67.90 | 102.4 | 57.49 |
| | 8 | 158.7 | 73.77 | 0 | 26 | 5 | 173.4 | 67.90 | 82.4 | 57.49 |
| | 9 | 172.3 | 81.75 | 0 | 26 | 5 | 193.4 | 67.90 | 62.4 | 57.49 |
| | 10 | 185.8 | 91.50 | 0 | 26 | 5 | 213.4 | 67.90 | 42.4 | 57.49 |
| | 11 | 201.5 | 98.25 | 0 | 28 | 3 | 217.9 | 86.73 | 48.0 | 64.37 |
| | 12 | 218.7 | 102.48 | 0 | 30 | 1 | 222.6 | 101.87 | 102.0 | - |
| Mean | | 144.6 | 74.56 | 0.0 | 24.5 | 6.5 | 150.0 | 62.61 | 93.2 | 66.20 |
| MST | 1 | 97.2 | 75.53 | 0 | 19 | 12 | 75.1 | 35.42 | 132.3 | 106.37 |
| | 2 | 102.8 | 68.86 | 0 | 21 | 10 | 86.8 | 42.52 | 136.5 | 99.66 |
| | 3 | 109.9 | 64.85 | 0 | 21 | 10 | 106.8 | 42.52 | 116.5 | 99.66 |
| | 4 | 117.3 | 65.80 | 0 | 22 | 9 | 121.2 | 49.06 | 107.7 | 98.84 |
| | 5 | 126.4 | 68.68 | 0 | 24 | 7 | 129.9 | 60.55 | 114.3 | 96.47 |
| | 6 | 138.3 | 70.40 | 0 | 25 | 6 | 144.5 | 65.13 | 112.5 | 91.54 |
| | 7 | 151.7 | 72.69 | 0 | 26 | 5 | 158.8 | 70.07 | 114.4 | 82.94 |
| | 8 | 166.2 | 75.32 | 0 | 27 | 4 | 172.8 | 75.60 | 121.8 | 64.69 |
| | 9 | 181.0 | 79.58 | 0 | 27 | 4 | 192.8 | 75.60 | 101.8 | 64.69 |
| | 10 | 195.9 | 85.82 | 0 | 27 | 4 | 212.8 | 75.60 | 81.8 | 64.69 |
| | 11 | 210.7 | 93.64 | 0 | 27 | 4 | 232.8 | 75.60 | 61.8 | 64.69 |
| | 12 | 227.1 | 99.09 | 0 | 29 | 2 | 236.2 | 95.73 | 95.5 | 30.41 |
| Mean | | 152.0 | 76.69 | 0.0 | 24.6 | 6.4 | 155.9 | 63.62 | 108.1 | 80.38 |
| PCM | 1 | 31.8 | 64.76 | 9 | 5 | 17 | 18.6 | 11.55 | 52.5 | 82.17 |
| | 2 | 27.0 | 58.79 | 11 | 14 | 6 | 12.4 | 15.30 | 110.7 | 97.63 |
| | 3 | 23.8 | 54.33 | 19 | 6 | 6 | 22.7 | 21.71 | 100.3 | 89.76 |
| | 4 | 18.5 | 46.01 | 10 | 16 | 5 | 7.9 | 5.71 | 89.6 | 90.20 |
| | 5 | 16.7 | 40.68 | 9 | 18 | 4 | 8.5 | 6.12 | 91.0 | 87.94 |
| | 6 | 14.3 | 35.35 | 18 | 10 | 3 | 14.9 | 9.16 | 98.0 | 77.54 |
| | 7 | 10.9 | 31.28 | 20 | 8 | 3 | 11.3 | 8.22 | 82.3 | 74.84 |
| | 8 | 9.5 | 27.10 | 21 | 8 | 2 | 15.3 | 10.17 | 86.0 | 86.27 |
| | 9 | 6.6 | 19.54 | 21 | 9 | 1 | 10.9 | 6.09 | 107.0 | - |
| | 10 | 6.3 | 22.87 | 20 | 9 | 2 | 7.2 | 6.24 | 65.5 | 86.97 |
| | 11 | 4.3 | 16.01 | 25 | 5 | 1 | 9.2 | 8.14 | 87.0 | - |
| | 12 | 3.9 | 12.32 | 23 | 7 | 1 | 7.6 | 4.58 | 67.0 | - |
| Mean | | 14.5 | 35.75 | 17.2 | 9.6 | 4.3 | 12.2 | 9.42 | 86.4 | 64.45 |

The limitation of the methods is the ability to reduce the average tardiness, which is to some extent due to the capacity of some of the machines in the system. From the table, it is noticed that though the number of tardy jobs is reduced under the proposed methods, the average tardiness does not decrease in some cases. Efforts will be made to improve this aspect in our future work.

In summary, as online methods which are simple to be implemented, require little information to make a decision, and will not be disturbed by disturbances, the proposed methods are significantly better in performance than the existing online control rules and are likely to bring improvement in performance to the real factory.

## V. CONCLUSIONS

This study tries to solve the problem of completing the jobs in a production shop at the times when the jobs are wanted by the successor shop by using online control methods.

The methods are simulated by the data of a real-world factory and are proven to be effective and helpful in achieving the objective. In the tested data, a majority of the jobs achieved on-time completion and a minority of them slight early completion. Tardiness is only witnessed in the jobs which are difficult to be in time due to the capacity of some machines.

The proposed methods are applicable to complex shops such as flexible job shops or flexible flow shops. The effectiveness in job shops or flow shops needs to be confirmed. The ideas involved such as forecasting the future waiting time of a job, ordering the jobs in a reasonable sequence in the processing and giving priority to the jobs that are close to their final operations may also be helpful to general production control problems.

As up to now, there have been few studies developing control methods for the objective of on-time job completion, this study has made a little step forward in filling up this void. But many relevant problems are still unsolved and methods proposed remain to be improved, more research attentions are deserved in this area where values can be created for efficiency along the production line or the supply chain.

## REFERENCES

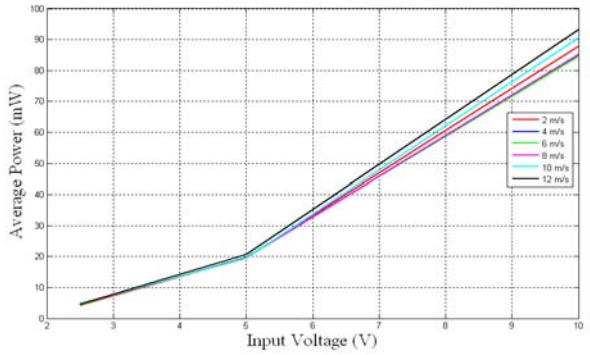
[1] M. Ham, Y. H. Lee, and S. H. Kim, "Real-time scheduling of multi-stage flexible job shop floor," *International Journal of Production Research*, vol. 49, pp. 3715-3730, 2011.

[2] T. C. Chiang and L. C. Fu, "Using dispatching rules for job shop scheduling with due date-based objectives," *International Journal of Production Research*, vol. 45, pp. 3245-3262, 2007.

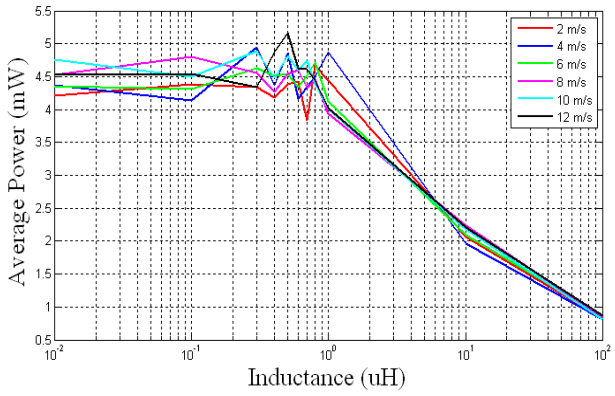
[3] T. Ohno, *Just-in-time for today and tomorrow*, Productivity Press, ISBN 978-0-915299-20-1, 1988.

[4] A. W. Mackelprang and A. Nair, "Relationship between just-in-time manufacturing practices and performance: A meta-analytic

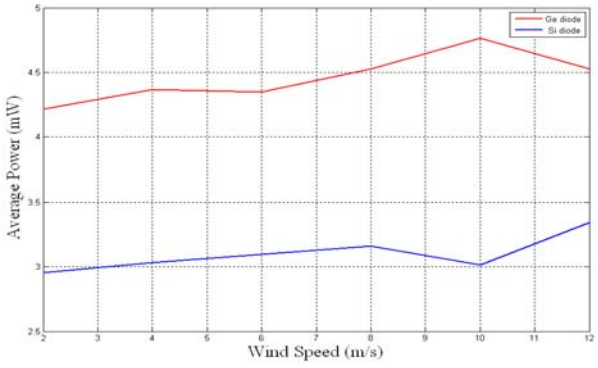
Fig.9 (b) indicates that the power increases when the rpm increases, where the maximum output power generated is 4.79 mW at 8 m/s when 0.1 µH inductor is used in transferring energy. This suggests there will be less power harvested when using high inductor values. Fig.9 (c) illustrates the variations of the generated power with the type of harvesting diode at various wind speeds. Higher power can be produced by reducing the forward voltage of the diode which is connected in series to the battery. As a result, a germanium diode with a forward voltage of 0.3 V is more suitable for this harvester circuit. Fig.9 (d) shows the influence of the battery capacitance on the output power at wind speed of 2 -12 m/s. The suitable capacitor for this circuit is between 1mF – 1 F. Higher capacitance values give unstable output power. This instability might be due to the big difference between the capacitance of $C_{var}$ and $C_b$ and the low capacitance values, give a low output power. Fig. 9(e) demonstrates the effect of varying the load resistance on the generated power at various wind speed. The results shows that the maximum power generated at resistance between 0.1 -1.0 Ω.



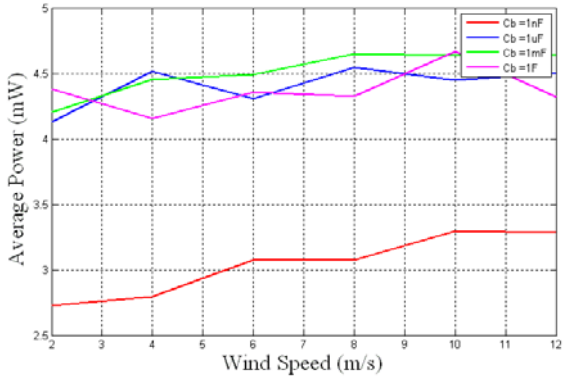

(a)



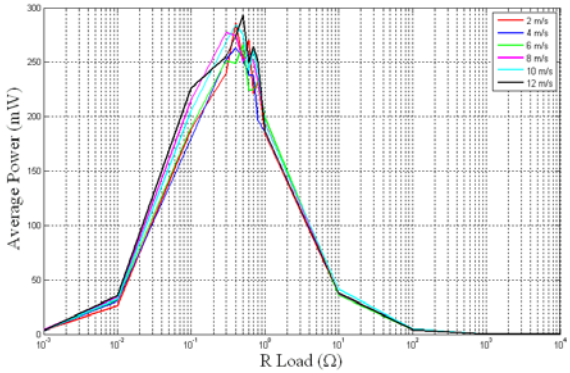

(b)




(c)




(d)




(e)

Fig. 9. Generated power under varying conditions for the harvester in Fig. 5 (a) Generated power with varying input voltage. (b) Generated power with varying inductance. (c) Generated power with two diode types. (d) Power variation with respect to wind speed at varying capacitance. (e) Power variation with respect to load resistance at varying wind speed.

TABLE II: THE POWER OUTPUT AT VARIOUS WIND SPEED

| Wind speed (m/s) | Cycle time (ms) | Output power (mW) | Energy gain (µJ) |
|---|---|---|---|
| 2 | 363.3 | 4.372 | 655.6 |
| 4 | 181.6 | 4.456 | 499.5 |
| 6 | 121.1 | 4.488 | 356.6 |
| 8 | 90.8 | 4.650 | 275.8 |
| 10 | 72.6 | 4.761 | 219.2 |
| 12 | 60.5 | 4.635 | 157.0 |

### *B. Optimization Results and Discussion*

Table II compares the power output and energy gain for various wind speed. The harvester model was tested at L = 0.01 - 0.1 µH, Diode voltage is 0.3 V, $C_b$ = 1 m F and $V_{in}$ is 2.5 V. The maximum harvested energy occurs at minimum wind speed. However, maximum harvested power occurs at maximum wind speed. The energy gain given in Table II can be worked out by multiplying the instantaneous power with the time. At wind speed of 2 m/s, the energy gain is 655.6 µJ for the time period of 363.3 ms. However, at wind of 12 m/s, the energy gain is 157 µJ for cycle time of 60.5 ms. the proposed harvester can complete 1 cycle within 363.3 ms at wind speed of 2 m/s and 6 cycles at wind speed of 12 m/s. Therefore the total harvested energy at higher wind speed is more than the energy gained at low wind speed for the same indicated cycle time. Fig .10 shows the harvested power and energy gain for wind speed of 10 m/s. The figure illustrate that the energy rises with time. Over the cycle time of 72.6 ms,