Adaptive Extensions to a
Multi-Branch Run-to-Run Controller for Plasma Etching

James R. Moyne, Nauman Chaudhry, and Roland Telfeyan

University of Michigan
Display Technology Manufacturing Center
Ann Arbor, MI 48109-2108
email: moyne@eecs.umich.edu

Abstract

Fuzzy logic and database learning mechanisms have been incorporated into a generic plasma etching run-to-run controller, resulting in a very dynamic, adaptable and robust system. The system features an Applied 8300 RIE controlled by a Techware II equipment controller. A TCP/IP connection links this equipment controller to the run-to-run controller residing on a SUN. The run-to-run control environment is generic in that the basic control framework and controller development results are applicable to VLSI manufacturing in general. The controller is multi-branch as it utilizes multiple algorithms in complementary fashion to achieve process optimization and control. The current implementation utilizes three branches: (1) a linear approximation control algorithm, (2) an optimization algorithm that utilizes (real-time) data collected in-situ to determine optimal run-to-run process parameter settings, and (3) a statistical optimization algorithm that utilizes run-to-run data. We have extended the controller to accommodate an automated branch selection process that utilizes fuzzy logic to incorporate process engineer as well as optimization and control algorithmic knowledge. We have also extended the controller to adapt to unforeseen events through utilization of a learning mechanism; this mechanism detects these unforeseen events, intelligently queries the process engineer, guides the engineer through the development of an event servicing scheme, and incorporates this new knowledge into its control knowledge base so that the event may be serviced automatically in the future. Implementation results of the controller (in the control of the etcher) confirm the robust control capabilities in the face of process shift and drift, and recipe change.

1.0 Introduction

Achieving optimization and control of plasma processes for purposes of maintaining acceptable yield in the face of smaller feature sizes and more complicated processes has proven to be an elusive task. The realization that the development of a comprehensive predictive physical process model set is probably years away has resulted in a general shift towards control that is based on empirical information (e.g., statistical process control (SPC) and neural networks) [2, 5, 7, 11]. In addition the lack of a critical mass of sensors and actuators for plasma process monitoring and

control has forced the utilization of innovative control techniques such as virtual sensors and multi-level hierarchical control [5, 7, 11]. Among the multiple process control levels, it is clear that run-to-run (R2R) control enjoys a higher level of maturity than time critical or real-time control due mainly to better sensing and actuation capabilities and the availability of a number of sequential control methods [7, 9, 12]. However all levels of plasma process optimization and control are far from a level of maturity required for industry-wide adoption and are thus receiving considerable attention as research topics in both industry and academia.

At the R2R control level a number of problems in implementation of results has hindered the wide-spread adoption of this type of control of plasma processes. First and foremost, much of the research in R2R optimization and control ends with proof of concept, which does not necessarily lead to adoption. The technology must be developed to be generic and portable to the many facets of plasma processing and indeed semiconductor processing in general. Second, in analyzing the alternatives, it appears that none of the available sequential control algorithms cover the entire spectrum of process optimization and control [7, 12]. A more robust controller framework is needed, i.e., a framework that supports the complementary utilization of a number of sequential optimization and control algorithms. This framework must contain a mechanism that can capture the state of the system after each process run and determine which of the available control algorithms is best suited to the process. Third, as both the plasma process and the state-of-the-art of control of the process are moving targets, R2R control must be adaptive to both the process and the technology. For example in many cases a R2R controller must be able to service unforeseen process events without taking the process off-line. Also, the controller must be able to adapt "on-the-fly" to incorporate additional process control knowledge and technology (e.g., the addition of a control algorithm "branch") as it becomes available.

In this paper we describe a generic, dynamic, adaptable and robust system for run-to-run control, focusing on recent enhancements in the area of utilization of fuzzy databases. This description builds on previous presented in Moyne, et. al. [7]. Specifically, following this introduction, the logical view of the R2R control framework and environment is introduced in Section 2. The controller design is then presented in Section 3; this presentation includes a description of multi-branch control, and learning mechanism framework components. The fuzzy database component of the controller is discussed in detail in Section 4; this component provides a method for the complementary utilization of R2R control algorithms, as well as on-line adaptation to new algorithms and control knowledge. Following this discussion, the results of the application of the system to the control of a plasma process are presented. We conclude the paper with a discussion of potential future research and development directions.

2.0 Control Environment

The R2R control framework design described in this paper is part of a multi-level control system that includes real-time equipment and process control as well as pseudo-real-time process control components operating in conjunction with the sequential control component [5, 7, 11]. A typical implementation of this multi-level control system is illustrated in Fig. 1. Note that the process control problem in this case is decoupled into three distinct components: in-situ process factory control, in-situ process control, and discrete (R2R) product control. In the ideal case these three

control loops operate concurrently in a hierarchical feedback fashion as shown. Note also that in the ideal factory control environment this multi-level process control system described is a component of a higher level factory control system that provides for the process integration and interprocess control.

As illustrated in Fig. 1, R2R or supervisory control is the highest level of control dedicated to a single process. This type of discrete control utilizes process and equipment data collected ex-situ along with historical knowledge of the process and equipment to suggest process recipe modifications so as to maintain or better achieve process output target values in subsequent process runs. The control feedback is accomplished by suggesting target output values to the next lower level of control. In the absence of lower level control, the R2R feedback would consist of suggestions for equipment input parameter settings (see Fig. 1).

The role of R2R control in this multi-level process control scheme is to a large extent dependent on the properties of lower layers of control. Specifically, it is dependent on design goals and requirements as well as the state-of-the-art in design and implementation at each level. As an example, it is generally accepted that R2R control technology is much more mature in this arena than that of direct equipment and process control [7]. Therefore near-term development of a generic R2R control system must not be dependent on the existence of equipment or process (i.e., in-situ) control. Eventually, however, as in-situ control reaches maturity, the generic R2R system should provide a migration path for the incorporation of the additional control capability.

A review of the role of R2R control in the process control scheme along with the state-of-the-art of sequential control in this arena reveals a number of design requirements for the R2R controller. These requirements are summarized in Table 1; note that we have described these requirements in detail in previous work [7].

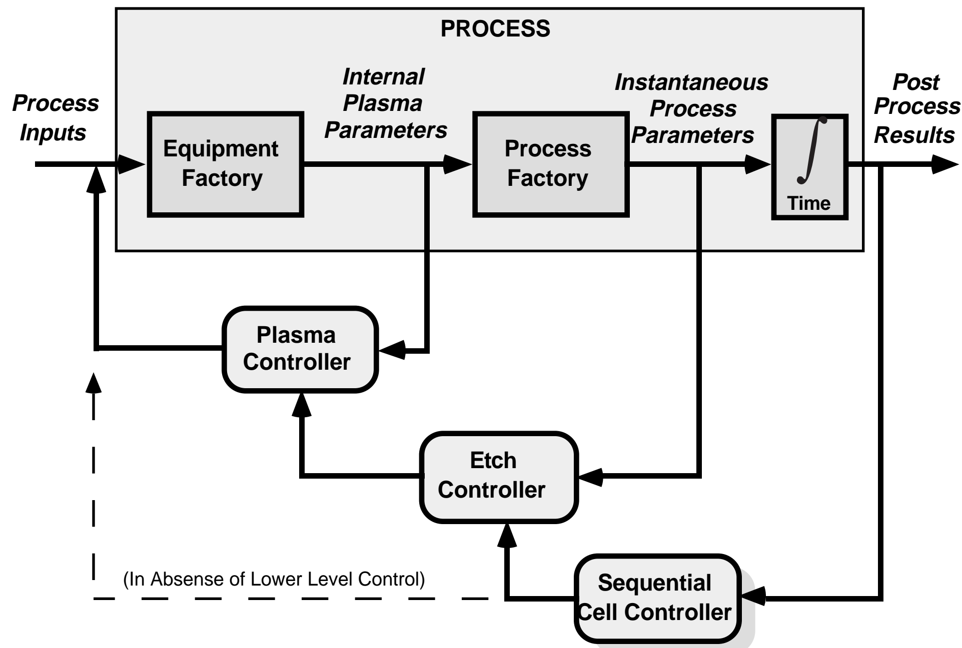


Figure 1: Multi-level Hierarchical Control System

1.)	Generic with respect to software, hardware, communication protocols, and process and equipment being controlled.
2.)	Capability to incorporate existing hardware and software components as necessary.
3.)	Capability to provide R2R control with or without lower level (in-situ) control.
4.)	Dynamic operation with respect to process drift and shift, and target shift.
5.)	Capability for in-operation updating of control scheme.
6.)	Accommodates existing software, hardware and communication standards.
7.)	Capability for utilization of a multitude of control and optimization branches in a complementary fashion.
8.)	Capability for incorporation of process and technology knowledge on-line.

Table 1: Generic R2R Control System Design Requirements

3.0 Controller Design

We have realized a R2R control design that conforms to the design requirements summarized in Table 1. A schematic of the two major design components, namely the enabling mechanism and the R2R optimization and control algorithm, are given in Fig. 2. The following is a brief description of these system components.

Generic Cell Controller

The Generic Cell Controller (GCC) is an enabler for discrete control systems, i.e., it serves as the "glue" to coordinate the information flow between software modules so as to achieve a coordinated objective such as R2R control [7,8]. The GCC mechanism utilizes a relational database as opposed to procedural code to store the sequential control information that is used to determine the interaction of the components of the control system. Thus the database contains all controller information for the sequential intelligent routing between the various modules of a sequential controller. For this reason the GCC is capable of supporting complex and dynamic routing schemes that are characteristic of many R2R control systems. Further, due to GCC database interaction specifications, a very high degree of modularity is established with GCC applications. This results in both high portability and transferability of software, and a capability to easily and

arbitrarily incorporate commercially available R2R control components into the system. A unique feature of the GCC is its ability to recognize and adapt to unforeseen control events "on-the-fly"; the GCC guides the process engineer through the development / learning of a scheme for the servicing of the event [8].

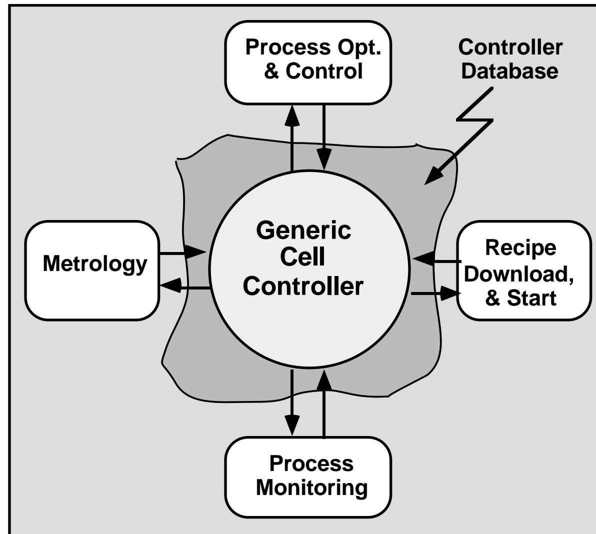
Process Optimization and Control Scheme

The process optimization and control scheme (illustrated in Fig. 2) includes a multi-branch process control and optimization method for R2R control [7]. Note that the scheme is enabled by a GCC implementation. Each branch consists of a single optimization or control algorithm (e.g., a commercially available algorithm). These branches are utilized in complementary fashion to enhance the robustness of the controller. In order to achieve complementary utilization of these algorithms, a mechanism must be devised to identify which combination of the available process analysis branches should be invoked so that the (statistically/heuristically) optimal or "best" advice on the process recipe will result. Further, after the process has been analyzed by the one or more indicated algorithm branches, a "best" set of advices (recipe) must be derived that results from comparing the weighted advices of the selected branch algorithms. This "best" recipe is the main output of the R2R controller. Such a mechanism is discussed in detail in the following section.

4.0 Branch Selection Algorithm

As stated previously, a necessary component of a R2R controller fully meeting the design requirements listed in Table 1 is a branch selection algorithm and associated advice weighting mechanism. There are a number of requirements placed on the branch selection algorithm by the R2R control paradigm. First, the knowledge of the state of the process that could be utilized to select R2R control algorithms is diverse, often times vague, and generally difficult to capture in crisp form. The same can be said of the knowledge pertaining to the domain of applicability of each branch algorithm. Indeed this knowledge is also vague. Further, as there is no common taxonomy for defining the domain of applicability, it is difficult to combine the knowledge into a deterministic knowledge-base.

A branch selection algorithm has been developed that satisfies the aforementioned requirements; the algorithm utilizes *fuzzy logic* applied to a knowledge base to determine branch selection [3, 4]. A review of the requirements of the branch selection algorithm and the available methods for its implementation reveals that fuzzy logic is indeed an ideal mechanism for the development and use of the required knowledge base [3, 6, 14]. Specifically, the fuzzy system that has been developed is attractive for many reasons. First, it makes effective use of vague or non-exact information in conjunction with deterministic information, and it can provide suggestions with a limited amount of knowledge. This is important because, in many cases, the available knowledge on branch selection for R2R control is somewhat vague and limited. Second, the developed fuzzy system effectively captures knowledge in "human language" format (a form in which much of R2R process control knowledge exists). Third, the system is capable of suggesting the "best" alternative(s) in situations where there may be many viable solutions. This is important because in many cases the domain of applicability of R2R algorithm branches overlap. Finally, the developed system can relate degrees of confidence with suggested solutions. Thus it inherently provides a mechanism for the weighting of advices from each branch invoked for a particular process run.



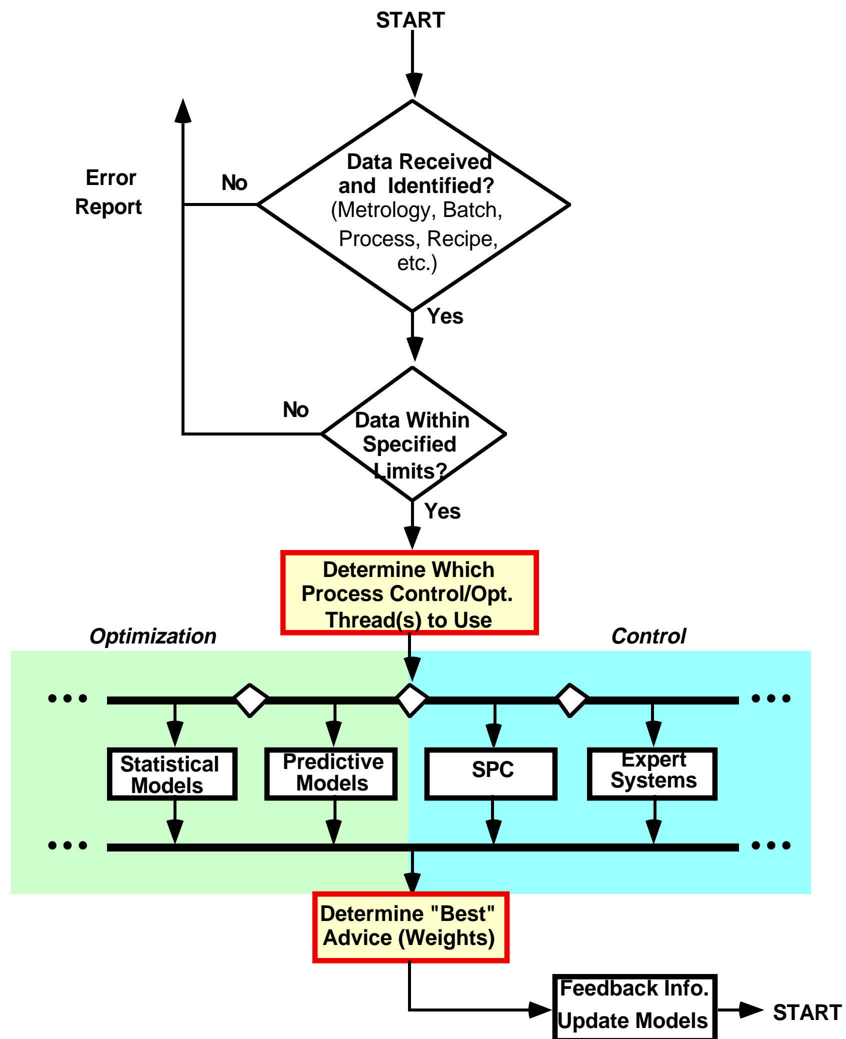


Figure 2: Control Enabling Mechanism and Optimization and Control Algorithm

The fuzzy mechanism developed incorporates both fuzzy and non-fuzzy knowledge into a data knowledge base. This knowledge base is incorporated / linked into the database of the GCC R2R control enabler, and the resulting system is thus able to enforce routing information relating to which control thread(s) to invoke for a particular process run (see Fig. 2) [4, 8]. Thus the database contains a schema for the storage of fuzzy and non-fuzzy rules as well as the interaction with the GCC database. It also contains a "fuzzifier" that categorizes process run data as necessary so that it may be utilized by the available rules [4].

The rule syntax allows expression of rules that advise *for* or *against* an action. As an example, Fig. 3 is an illustration of a valid rule base. The rule base contains rules that relate the usefulness of one of two algorithms (a linear approximation control algorithm, and a quadriatic approximation optimization algorithm) to the process error. Note that each rule contains a predicate, operator, action and certainty factor (a number between zero and one indicating the confidence or believability of the rule).

In order to handle information that is somewhat contradictory (e.g., rules 2 and 4 in Fig. 3) we utilize a method introduced by [1]. With this method, all rules associated with a particular action (i.e., the action of choosing a particular branch) are partitioned into two sets, those recommending *for* the action and those recommending *against* the action [4]. For each of these sets a *confidence* is derived (a number between zero and one) by applying fuzzy set theory. Thus an *Upper Confidence* level advising *for* an action and a *Lower Confidence* level advising *against* an action are both derived.

After both confidences are derived, we then provide the following technique for making a decision of branch selection [4].

1.) Upper and Lower Confidence levels computed are combined into a tuple (X_1, X_2), where X_1 represents the degree of support for an action, and X_2 represents one minus the degree of support against an action ($0 \leq X_1, X_2 \leq 1$).

2.) The tuple is then plotted as shown in Fig. 4. This graph is a two-dimensional representation of the support associated with an action. Thus, the tuple associated with an action plotted on the graph depicts the degree to which the action is confirmed or refuted, and the degree to which the rule set associated with the action is contradictory or supplying a low amount of information. In Fig. 4, rules strongly advise for choosing algorithm "A" and against algorithm "B". Rules associated with algorithm "C" are contradictory (i.e., some are expressing strong support for the algorithm while others are expressing strong support against its utilization). There is little confidence in the knowledge base information associated with algorithm "D". Placement of points "E" and "F" indicate intermediate levels of confirmation and rejection support.

3.) Steps 1 and 2 are applied to rule sets associated with other actions (that suggest other branches).

4.) The graph is partitioned into three regions as shown in Fig. 4: A triangular region of strong confirmation recommendation, a triangular region of strong rejection

recommendation, and the remaining area representing weaker recommendation. Note that the two partitioning lines are lines of equal confidence.

5.) A rule is applied to the graph to determine which action(s) to take, i.e., which optimization and / or control algorithm(s) to invoke. An example of such a rule might be:

if (there is at least one action in the strong confirmation region)

then (take all actions in the strong confirmation region)

else *if* (there is at least one action not in the strong rejection region)

then (take the action closest to the strong confirmation region)

else (take no action)

Note that if this rule were applied to the event depicted in Fig. 4, then algorithm "A" would be invoked for the current process run.

In summary, the branch selection method developed utilizes fuzzy logic theory to recommend optimization and / or control branches to be invoked for a particular process run. The method is flexible in that it supports fuzzy rules, such as rules that might be attained from the process engineer, as well as non-fuzzy rules. Further the method is adaptive as it can incorporate new rules (relating to existing *or* new branch selection decisions) "on-the-fly". This property combined with the GCC learning mechanism capability results in a R2R control framework that is very dynamic and adaptable.

Rule 1: if error > 3 sigma, then quadOpt, 1.0

Rule 2: if error > 3 sigma, then not linearApp, 1.0

Rule 3: if error < 3 and error > 2, then quadOpt, 0.7

Rule 4: if error < 3 and error > 2, then linearApp, 0.4

Rule 5: if error < 2 and error > 1, then quadOpt, 0.5

Rule 6: if error < 2 and error > 1, then linearApp, 0.7

Rule 7: if error < 1, then not quadOpt, 1.0

Rule 8: if error < 1, then linearApp 1.0

Figure 3: Fuzzy Rule Base for a Two-Branch Controller

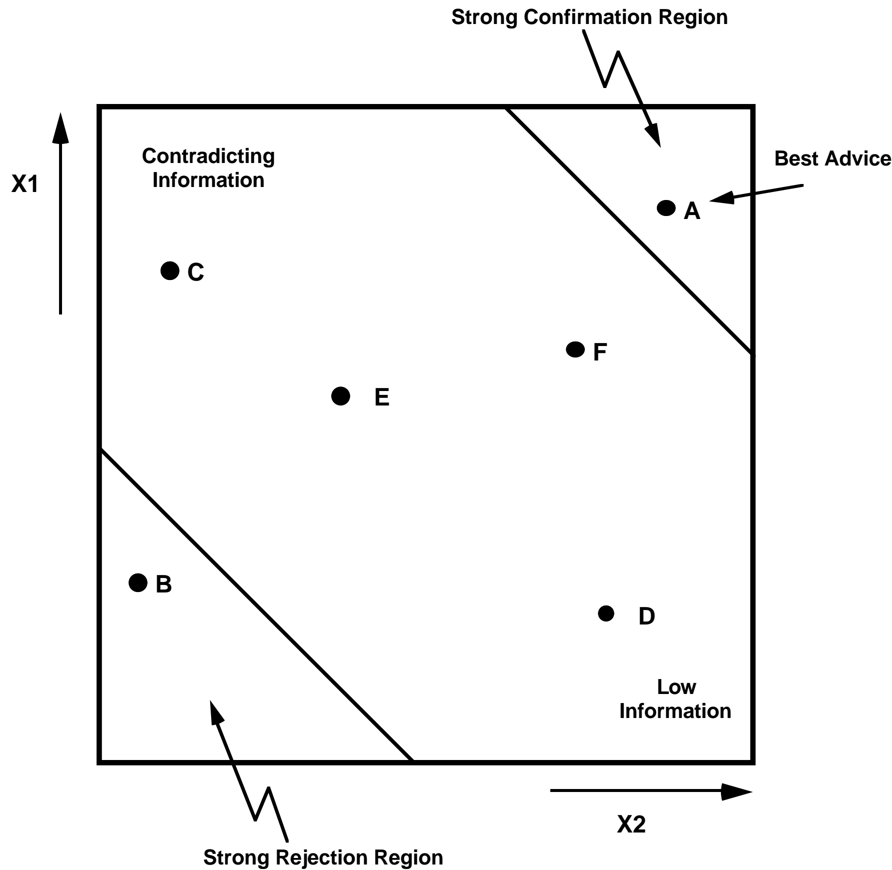


Figure 4: Two Dimensional Representation of Action Support

5.0 Application to Control of Plasma Processing

A feasibility implementation of the R2R framework has been developed for the optimization / control of a plasma etching process in an Applied Materials 8300 Reactive Ion Etcher. The R2R controller was developed on a SUN Microsystems SPARC 10 workstation. The GCC database (including the fuzzy branch selection knowledge base) was developed using an ORACLE RDBMS and interacts with the rest of the controller via a Structured Query Language (SQL) interface [10]. The R2R controller communicates with a Techware II equipment controller via a TCP/IP over Ethernet communication link [13].

Experiments conducted for this case study involved the etching of a simple 3-layer PolySilicon / SiO_2 / Si wafer. Recipe targets were etch rate and target over-etch, while adjustable recipe parameters were oxygen flow and etch time. For purposes of demonstration of control, the system was perturbed by introducing oxygen into the system (simulating an oxygen leak) and changing the etch rate target parameter (i.e., adjusting the recipe). The optimization / control scheme that was utilized incorporates three branches: (1) a multiple-input-multiple-output (MIMO) linear approximation control algorithm [12], (2) an optimization algorithm that utilizes (real-time) data

collected in-situ to determine optimal run-to-run process parameter settings [7], and (3) a statistical optimization algorithm that utilizes run-to-run data [9].

The utilization of all three threads in a single control scheme has not yet been implemented, however implementations of two threads have been applied to the process with successful results. As an example, Fig. 5 illustrates successful R2R control of the process using a MIMO linear approximation control algorithm branch after optimizing the process with the algorithm that utilizes real-time data to determine optimal run-to-run process parameter settings [7]. Note that the controller adjusts the process rapidly to compensate for both target shifts (etch rate target changed after run #8) as well as process perturbations such as sensor drift (oxygen flow sensor perturbed by .2 sccm after run #3). In this case, the knowledge base indicated that the MIMO control algorithm was appropriate to both (shift and drift) compensation tasks as well as control when neither drift nor shift was evident. As a second example, the simple fuzzy rule base for a multi-branch controller introduced (earlier) in Fig. 3 consists of the same MIMO linear approximation control algorithm along with a statistical optimization algorithm that utilizes run-to-run data. This control scheme has been successfully applied to the control of a planarization process.

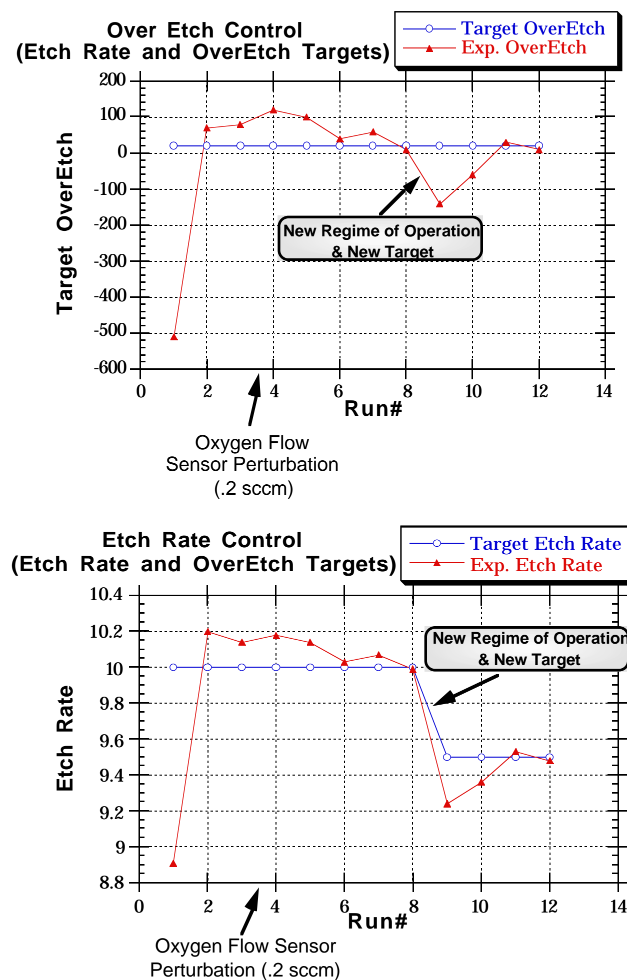


Figure 5: Illustration of R2R Control Utilizing a Linear Approximation Control Algorithm Branch (Optimization Via Real-Time Data Collect In-Situ)

6.0 Conclusions

In this paper we have described a generic, dynamic, adaptable and robust system for run-to-run control. This description builds on previously reported work, focusing on a recent enhancement to the system, namely the design, development and incorporation of a fuzzy knowledge data base that provides for the complementary utilization of optimization and control algorithms in the R2R control system. We have defined a mechanism for the incorporation of fuzzy and non-fuzzy rules into the data base, and a graphical-based method of analyzing these rules and concluding which optimization and / or control "branches" should be invoked for a particular process run. Experimental results indicate the advantages in robustness of multi-branch control over single branch control, and illustrate the effectiveness of a fuzzy database in capturing knowledge relating to the branch decision making process.

In the near future we hope to expand the fuzzy knowledge base and apply the run-to-run control system in both a simulated and real environment in order to: (1), validate and expand the branch selection knowledge base, (2), better identify the domain of operation of the various optimization and control algorithms especially with respect to the plasma process, (3), identify combinations of optimization and control threads that characterize ideal control solutions, (4), identify deficiencies within the stated domain of operation of existing algorithms, as well as "holes" in the optimization / control domain where algorithms should be expanded or new algorithms developed, and (5), provide quantative data comparing open loop operation, single branch closed loop control, and various schemes of multi-branch closed loop control. In this way we hope to further enhance the robustness of the R2R control framework and obtain a better understanding of role of R2R control in the semiconductor manufacturing control framework.

7.0 Acknowledgments

The authors gratefully acknowledge the contributions of Hossein Etemad, Jai Jacob and Saqib Syed toward developing the R2R control implementation software, Prof. Elke Rundensteiner toward developing the fuzzy theory utilized in the controller, and Dr. Michael Elta for managing the control framework development program. This work is supported in-part by the Semiconductor Research Corporation and the Center for Display Technologies.

8.0 References

[1] J. Buckley and D. Tucker, "Second generation fuzzy expert system," Fuzzy Sets and Systems 31 (1989).

[2] Camporese, D., "Artificial Neural Networks for Real-Time Tool Controllers," *SRC RTTC Workshop*, (February 1991).

[3] N. Chaudhry, J. Moyne, E. Rundensteiner, "A Design Methodology for Database with Uncertain Data", *Proceedings of the Seventh International Working Conference on Scientific and Statistical Database Management*, Charlottesville, Virginia, (September 1994).

[4] N. Chaudhry, J. Moyne, E. Rundensteiner, "A Fuzzy Database for Run-to-Run Control of Semiconductor Manufacturing," (in preparation).

[5] M.E. Elta, H. Etemad, J.S. Freudenberg, M.D. Giles, J.W. Grizzle, P.T. Kabamba, P.P. Khargonekar, S. Lafortune, S.M. Meerkov, J.R. Moyne, B.A. Rashap, D. Teneketzis, F.L. Terry, Jr., "Applications of control to semiconductor manufacturing: reactive ion etching," *Microelectronic Processing '93 Proceedings*, Vol. 2091, September 1993.

[6] C. Lee, "Fuzzy logic in Control Systems: Fuzzy Logic Controller, Parts 1 and 2," IEEE Transactions on Systems, Man, and Cybernetics 20, 2 (March/ April 1990).

[7] J. Moyne, H. Etemad, M. Elta, "Run-to-run Control Framework for VLSI Manufacturing," *Microelectronic Processing '93 Conference Proceedings,* Monterey, California, (September 1993).

[8] J. Moyne and L. McAfee, Jr. "A generic cell controller for the automated VLSI manufacturing facility," *IEEE Transactions on Semiconductor Manufacturing*, Vol. 5, No. 2, pp. 77-87, May 1992.

[9] C.W. Moreno, "Self-Learning Optimizing Control Software," *ISA Proceedings*, June 1986.

[10] ORACLE Corporation, Red Bluffs, California.

[11] B. Rashap, P. Knargonekar, J. Grizzle, M. Elta, J. Freudenberg, and F. Terry, “Real-Time Control of Reactive Ion Etching: Identification and Disturbance Rejection,” *Proceedings IEEE CDC* (1993).

[12] E. Sachs and A. Hu, "Run by run process control: Combining SPC and Feedback Control," (submitted to) *IEEE Transactions on Semiconductor Manufacturing*, (October 1991).

[13] Techware Systems, Richmond, British Columbia, Canada.

[14] L. Zadeh, "Fuzzy Sets," Information and Control 9 (1965).