Research ID 880: Scheduling of Semiconductor Wafer Fabrication Facilities

2002 Project Review

Matt Carlyle, John Fowler, Esma Gel, Scott Mason, Lars Mönch, Michele Pfund, Oliver Rose, George Runger, Roland Sturm, Steven Brown

Arizona State University, Fraunhofer Institute – IPA, Naval Postgraduate School, Technical University of Ilmenau, University of Arkansas, University of Würzburg
# 880 Task Summary - Mason

<table>
<thead>
<tr>
<th>Number</th>
<th>Task/Activity</th>
<th>Task Leader</th>
<th>Deliverable(s)</th>
<th>Students</th>
</tr>
</thead>
<tbody>
<tr>
<td>880</td>
<td>Scheduling of Semiconductor Wafer Fabrication Facilities</td>
<td>John Fowler</td>
<td>Final Report (12/03)</td>
<td></td>
</tr>
<tr>
<td>880.001</td>
<td>Scheduling and Rescheduling Methodologies</td>
<td>Scott Mason</td>
<td>Annual Report 12/01 &lt;br&gt;Annual Report 12/02 &lt;br&gt;Annual Report 12/03</td>
<td>Song Jin &lt;br&gt;Peng Qu &lt;br&gt;Chris Wessels &lt;br&gt;Oliviana Zakaria</td>
</tr>
<tr>
<td>880.002</td>
<td>Subproblem Solution Procedures</td>
<td>Michele Pfund</td>
<td>Annual Report 12/01 &lt;br&gt;Annual Report 12/02 &lt;br&gt;Annual Report 12/03</td>
<td>Hari Balasubramanian &lt;br&gt;Amit Gadkari &lt;br&gt;Shari Murray &lt;br&gt;Amit Devpura &lt;br&gt;Heiko Niedermayer &lt;br&gt;Rüdiger Martin</td>
</tr>
<tr>
<td>880.003</td>
<td>Statistical Operations Control</td>
<td>Esma Gel George Runger</td>
<td>Annual Report 12/01 &lt;br&gt;Annual Report 12/02 &lt;br&gt;Annual Report 12/03</td>
<td>Andy Burhanuddin &lt;br&gt;Cem Vardar &lt;br&gt;Andre Pfeuffer</td>
</tr>
<tr>
<td>880.004</td>
<td>Testing, Comparison, &amp; Implementation Issues</td>
<td>Oliver Rose</td>
<td>Annual Report 12/01 &lt;br&gt;Software Env. 12/01 &lt;br&gt;Annual Report 12/02 &lt;br&gt;Software Env. 12/03</td>
<td>Florian Holzinger &lt;br&gt;Kai Hübner &lt;br&gt;Bosun Kim &lt;br&gt;Detlef Pabst &lt;br&gt;Torsten Schiele &lt;br&gt;Maryam Zehtaban &lt;br&gt;Jens Zimmermann</td>
</tr>
</tbody>
</table>
880 Project Management

- Project Website
  - blackboard.dlt.asu.edu

- Weekly Site Meetings at ASU, IPA, TUI, UA, and UW
- Weekly Task Meetings for Tasks 1, 2, and 3
- Bimonthly Task Meetings for Task 4

- Monthly Teleconferences With all PI’s
- Monthly Teleconferences With Liaisons and PI’s
880 Year 1 Executive Summary

This year has been very productive for our project focused upon developing and testing a shifting bottleneck based approach to scheduling wafer fabs. In particular:

- We have coded the shifting bottleneck algorithm developed by Mason et al. in C++ and have solved a cycling problem which can occur in the original algorithm (task 1).

- We have also developed and tested a subproblem solution procedure (SSP) module template for use with the shifting bottleneck code. Preliminary SSPs have been developed for reticle management, parallel machines, and parallel batch machines (task 2).

- We have identified potential triggers that indicate when rescheduling might be necessary (task 3).

- We have developed an ASAP based modeling environment to test the performance of the shifting bottleneck and other scheduling approaches. Preliminary tests with simple dispatching rules have been successfully completed (task 4).
880 Year 2 Executive Summary

This year has been very productive for our project focused upon developing and testing a shifting bottleneck based approach to scheduling wafer fabs. In particular:

– We have explored the incorporation of automated material handling systems in our Shifting Bottleneck algorithm and investigated various approaches for black boxing non-critical tool groups in the fab (task 1).

– This year was dedicated to the development of Subproblem Solution Procedures for tool groups with auxiliary resources (reticle management), single machine, parallel machines, batching machines and cluster tools. We developed an interface with the SBH to create a SSP library that is readily accessible from the development environment, is fully compatible with the current development architecture, and is easy to maintain and upgrade (task 2).

– The last year has been focused upon evaluating possible event and data triggers that might be used to decide whether it might be desirable to generate a new schedule or take some other corrective action (task 3).

– This year was dedicated to refining/improving the ASAP Testing environment and the development of the data model as the interface between fab and scheduler. We customized ASAP model of MiniFab Model, MIMAC Test Bed Data Set 1 Model, and ISMT 300mm Fab Model to export/import data to/from scheduler via the data model (task 4).
2001 Annual Review Feedback for 880

- **Overall**
  - + Appreciate excellent student presentations
  - + Excellent management of complex project
  - > Look into supplier claim of products with these functions
    - Held series of meetings with Adexa – will restart now that we have a working prototype of system
    - Continue excellent relationship with Brooks
  - > Watch literature for promising algorithms
    - We continue to do this
2001 Annual Review Feedback for 880

- 880.001 Scheduling / Rescheduling Methodologies
  - Promising approach
  - Lots of publications
  - Need to provide for lot going through same stepper next time through
    - Added to list of requirements - will assess impact during Year 3
2001 Annual Review Feedback for 880

- 880.002 Subproblem Solution Procedures
- Glad to see a lot of deliverables projected

- > Clarify which decisions are made at sub-problem level, which at SBH level and what info is passed between the two levels
  - Added to yearly deliverable requirements

- > Integrate with PM scheduling project
  - 2003 task, but have held preliminary discussions with Emmanuel Fernandez

- > Any new deliverables for reticle management?
  - Current plan is to develop / evaluate new SSP’s for reticle management
2001 Annual Review Feedback for 880

- 880.003 Statistical Operations Control
  - + Like that you are leveraging info from disjunctive graph
  - + Good analysis and novel approaches
  - > May need a lot of education for end users (technical details)
    - Gave an update of this work at MASM 2002 and INFORMS San Jose and will include details in technical documentation (deliverable)
2001 Annual Review Feedback for 880

- 880.004 Testing, Comparison and Implementation
  - + Good progress – completed a lot of work on time
  - + Good communication

- > Would like data model presented in liaison meeting
  - Presented in March meeting

- > Please test all the common scheduling/dispatching rules
  - This is in the plan. Please let us know if there are other rules you want us to test.

- > Is CIM Framework a resource for the data models of both the fab emulator simulation and the scheduler in the simulation/test environment structure?
  - Currently being investigated by German team
880.001: Scheduling and Rescheduling Methodologies

Research Personnel:

- Task Leader: Scott Mason, UA
- Faculty:
  - Michele Pfund, ASU
  - John Fowler, ASU
  - Oliver Rose, UW
- Students
  - Chris Wessels, UA, BS 12/02 (SRC Research Opportunities for Undergrads), MSIE 5/04
  - Peng Qu, UA, MS 6/02, PhD 8/04
  - Song Jin, UA, MS 5/03
  - Oliviana Zakaria, UA, MSOR 12/03
- Liaisons
  - Sidal Bilgin, LSI
  - Wayne F. Carriker, Intel Corporation
  - Paul A. Flores, Intel Corporation
  - James Heskin, TI
  - Sarah Hood, IBM
  - Joanna Shear, Intel
  - Kwodle Prasad, Intel
  - Raja S. Sunkara, NSC
880.001: Scheduling and Rescheduling Methodologies

Primary Anticipated Results:

- Develop a Shifting Bottleneck-based approach for scheduling a wafer fab
  - Main scheduling approach
  - “Rescheduling” approach that improves suboptimal schedules in real time
- Investigate incorporation of automated material handling systems into the Shifting Bottleneck algorithm
- Investigate simplifying fab models by “black-boxing” non-critical areas of the fab
- Investigate incorporation of order release techniques into the scheduling framework
880.001: Scheduling and Rescheduling Methodologies

Task Description:

- **Year 1** - Implementing the overall shifting bottleneck approach for scheduling, and its preparation for integration into our testing environment described in Task 4. Once the main procedure is implemented, the remaining effort in this year will be the implementation of rescheduling.

- **Year 2** - Develop more advanced techniques, especially in the area of building the scheduling models. The most important of these will be the development of techniques to identify non-critical areas of the fab. This stage of Task 1 will also investigate the incorporation of automated material handling systems into the scheduling models.

- **Year 3** - Implement a module that can determine optimal order release policies. This will allow the scheduling algorithm (or possibly a suitable order release SSP, as an additional Task 2 exercise) to help determine the appropriate lot releases over the scheduling model horizon.
880.001: Scheduling and Rescheduling Methodologies

Task Deliverables:

- Development and implementation of the shifting bottleneck approach and the basic rescheduling methodology. Report (Year 1)

- Methods to combine tool groups into non-critical areas and abstract them out of the models. Report (Year 2)

- Incorporation of material handling issues into shifting bottleneck approach. Report (Year 2)

- Order release policies and integration into scheduling approaches. Report (Year 3)
880.001 Scheduling and Rescheduling Methodologies

Executive Summary

• Continued ASAP framework testing to find and resolve any errors in the Shifting Bottleneck heuristic.

• Extended Evaluation Test Suite (ETS) to ASAP environment, developing appropriate ASAP simulation models.

• Investigated approaches for “black boxing” non-critical machines/tool groups/areas of the fab.

• Investigated incorporating automated material handling systems within the Shifting Bottleneck framework.
Recent Accomplishments and Progress—Evaluation Test Suite

- Extended original Evaluation Test Suite (ETS) for C++ code to ASAP environment.
  - The ETS contains a set of standard model input files that will be used to automatically test both solution quality and code execution time in the future as additional enhancements are added to the core SB code.

- Recently added Shifting Bottleneck customization to ISMT 0.18μ 300-mm model so that we can test our scheduling approach with a larger, more realistic fab model.
  - Original customization included in Sematech’s release of this model has been removed to facilitate more rapid testing procedures.
Recent Accomplishments and Progress—Black Boxing

- Investigated approaches for “black boxing” non-critical machines/tool groups/areas of the fab
  - Our goal is to assess the trade-off between improved execution time and solution quality.
  - George Washington Carver Project Intern joined Arkansas team for six weeks (Joyce Beal) and helped develop single step and multiple step approaches.
  - Initial results demonstrated expected improvement in computation time, but solution quality results warrant further investigation into C++ code.
Blackboxing Overview

- Our Shifting Bottleneck (SB) procedure schedules wafer fabs with \( m \) tool groups (TGs) in a total of \( m \) iterations
  - "Most critical" TG scheduled at each iteration
    - \( m \) TG subproblems evaluated at iteration 1
    - \( m-1 \) TG subproblems evaluated at iteration 2, ...
  - A total of \( m(m+1)/2 \) subproblems evaluated

- Each subproblem must be formed, then evaluated
  - Subproblems = Solution time
“Black Boxing” Tool Groups

- In wafer fabs, all TGs do not possess the same criticality
  - For example, steppers are more important than microscopes

- As part of Task 1’s Year 2 research, we are developing approaches for
  - Identifying non-critical TGs in the wafer fab
  - Converting these non-critical TGs into simple process delays in the disjunctive graph
  - This will reduce solution time, as less TGs will exist in our problem—however, we need to understand the cost paid in solution quality associated with this improved solution time.
An Example

- (D)iffusion, (E)tch, (P)hoto, (M)icroscope
- Assume all $r_j = 0$, $p_j = 10$
What if Microscopes are non-critical?

- Remove all (M) nodes and adjust arc lengths
- Problem: Preceding TG now over constrained
A Better Approach

- Consolidate (M) nodes into Delays ("Z"), adjusting arc lengths as necessary
- Remove TG Z from SB (assume it has infinite cap)
Black Boxing—Other Issues

- While black boxing will indeed improve computation time, we need to see if/by how much solution quality degrades.

- Should batch processing tools EVER be deemed non-critical?

- Should simply the TG’s $p_j$ be counted as the delay, or some multiple of $p_j$?
Recent Accomplishments and Progress—Automated Material Handling Systems

- Investigated approaches for incorporating AMHS within the Shifting Bottleneck framework
  - Initial results demonstrated expected increase in TWT when AMHS is a near constraint. Additional nodes in disjunctive graph also increase solution time.
  - Further investigation with real fab data warranted to assess the criticality of the AMHS (i.e., can we black box it or not?).
Adding AMHS to the Disjunctive Graph

If $f_j$ is number of processing steps in job $j$’s process routing, graph with AMHS has $2 + n + \sum_{j=1}^{n} 2f_j$ nodes, rather than $2 + \sum_{j=1}^{n} f_j$ nodes for no AMHS case.
880.001: Scheduling and Rescheduling Methodologies

Future Plans and Directions

• Continue testing of scheduling and rescheduling methodologies within ASAP testing environment.

• Investigating methods for ensuring a lot can be required to go through same stepper at subsequent processing layers

• Investigate how to incorporate order release decisions into the approach.
880.002: Subproblem Solution Procedures

Research Personnel:

• Task Leader: Michele Pfund

• Faculty
  – Matt Carlyle
  – John Fowler
  – Scott Mason
  – Lars Mönch
  – Oliver Rose
  – Roland Sturm

• Students
  – Hari Balasubramanian, ASU, MS 5/02
  – Amit Devpura (Intel), ASU, PhD 8/03
  – Amit Gadkari, ASU, MS 5/02
  – Rüdiger Martin, UW, MS 6/02
  – Shari Murray, ASU, MS 12/02
  – Heiko Niedermayer, UW, MS 6/02
  – Detlef Pabst, UW

• Liasions
  – Manuek Aybar, TI
  – Sidal Bilgin, LSI
Primary Anticipated Results:

- Develop sub problem solution procedures
  - for specific tool groups and topologies
  - extend shifting bottleneck approach of task 880.001

- Important research issues
  - photolithography steppers and their reticle management
  - cluster tools
  - alternate subproblem objectives
880.002: Subproblem Solution Procedures

Research Description:

- **Year 1** – Develop SSPs for reticle management in photolithography. Adapt a network flow model developed by the PI’s under previous SRC funding. The procedure will schedule jobs and reticles and provide reasonable completion time estimates.

- **Year 2** – Develop special SSPs for cluster tools. Cluster tools provide a completely different modeling challenge than standard equipment. Extend models developed by Dümmler and Rose (U of Würzburg) and integrate them into our scheduling approach.

- **Year 3** – Investigate other performance measures besides total lateness. e.g., the schedules for certain machines may need to be operating parameter insensitive and require other performance measures. Develop SSPs for this purpose and integrate them into our scheduling framework.
880.002: Subproblem Solution Procedures

**Deliverables:**

- Report on reticle management models implementation. Report (Year 1)
- Report on cluster tools models implementation. Report (Year 2)
- Report on alternate performance measures implementation. Report (Year 3)
880.002: Subproblem Solution Procedures

- **Executive Summary:**

  This year was dedicated to the development of Subproblem Solution Procedures for tool groups with auxiliary resources (reticle management), single machine, parallel machines, batching machines and cluster tools.

  In this year, we have developed an interface with the shifting bottleneck heuristic to create a SSP library that is readily accessible from the development environment, is fully compatible with the current development architecture, and is easy to maintain and upgrade.
880.002: Subproblem Solution Procedures

• Year 1 Accomplishments and Progress
  
  – Literature review to find efficient heuristics for the different Subproblem Solution Procedures
  – First In First Out (FIFO) based reticle management code implemented in C++
  – Preliminary data model for SSP’s developed
  – BATCS code implemented in C++, handles most tool groups that do not require auxiliary resources and are not cluster tools
  – BATCS code integrated with the shifting bottleneck heuristic
880.002: Subproblem Solution Procedures

- **Year 2 Accomplishments and Progress**
  - SSP data structure defined and implemented
  - SSP library created for single, parallel, and batch machines
  - Genetic algorithm developed for:
    - cluster tools (competitive heuristics being developed)
    - parallel machines with due dates and unequal ready times
    - batch parallel machines with due dates and incompatible job families
  - Machine learning technique developed for determining k parameters for BATCS
  - Improved ATCs formulation to better incorporate ready times
  - Reticle management optimization – algorithm development & ASAP implementation definition
Subproblem Solution Procedures for Cluster Tools Prediction of Lot Cycle Times

Heiko Niedermayer
Introduction & Overview

- Our goal: Scheduling of cluster tools

- Simulator: CluSim
  by Dümmler/Schmid/Bohr (University of Würzburg)

- Simulation-based scheduling is very expensive:
  - The SB algorithm calls SSP several times
  - The SSP itself needs several simulation runs
  - Is it possible to avoid simulation?

- Are cluster tools just like parallel machines?
  No! Cycle times of lots processed in parallel are correlated.
Overlap & Cycle time

No overlap
Lot A
Lot B

Partial overlap
Lot A
Lot B

Complete overlap
Lot A
Lot B

Typical graph:

Lot A/B saturated

Lot A
Lot B
How lots can be handled

Given: n Recipes

- Single mode
  - cycle time: $O(n)$ simulation runs, CPU time and storage
  - behaves like a single machine

- Parallel mode – both loadlocks are only loaded and unloaded at the same time
  - cycle time: $O(n^2)$ simulation runs, CPU time and storage
  - a bit like batch machines, but with completely different characteristics

- Parallel mode – general case
  - cycle time: complete simulation of the scenarios or ???

That's what the rest of this talk is about.
Slow Down Factors

- Lot A is slowed down by Lot B

- Definition:
The slow down factor of lot A processed in parallel to Lot B is

\[
SDF(A, B) = \frac{CycleTime(LotA\_together\_with\_lots\_of\_type\_B)}{CycleTime(LotA\_alone)}
\]
Prediction of Slow Down Factors

Goal: predict slow down factor for lot A combined with lot B

Input data:
- Requested raw processing time for each chamber, handler, the bottleneck and their averages
  - Case 1: lot A is processed alone (SOLO)
  - Case 2: both lot types are processed in parallel (COMBI)
- Cycle Time of lot A being processed alone

Predictors (Selection):
- BRPT_ratio (COMBI Bottleneck time / SOLO Bottleneck time)
- Linear (normalized linear combination of the input)
- Ratio (ratio of two linear combinations of the input)
- RatioAB2A (linear combination of COMBI inputs / linear combination of SOLO inputs)
- Neural networks (inputs normalized)
Prediction of Slow Down Factors

Results:
- Ratio and the neural networks produced the best results
- Ratio generalizes well, even with only 20-40% of the patterns known
- Neural networks should be better able to handle more recipes than Ratio

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Average absolute error (study with 12 recipes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ratio</td>
<td>0.02</td>
</tr>
<tr>
<td>Neural networks</td>
<td>0.07</td>
</tr>
<tr>
<td>Linear</td>
<td>0.30 (Regression: &gt; 80% of variation is explained)</td>
</tr>
<tr>
<td>RatioAB2A</td>
<td>0.35</td>
</tr>
<tr>
<td>BRPT_ratio</td>
<td>0.40</td>
</tr>
</tbody>
</table>
Prediction of Lot Cycle Times

- Scenarios: n lots enter the cluster tool in a given sequence
- Cycle time prediction using the slow down factors from CluSim simulation study
- Results: for reasonable recipe combinations the predictions are very accurate (< 10 % error)
- Deciding if a lot combination is reasonable is possible since slow down factor predictions and other heuristics are good enough to detect good and bad lot combinations!
Problem Identification

• Realistic Reticle Requirements
  – 8 product family
  – 12 product types per family
  – From 16 to 27 layers per product
  
  **2064 reticles required**

• Realistic Photolithography Station
  – 16 machines
  – 12 storage slots per machine

  **192 reticle storage capacity**
Characteristics of Ideal Reticle Management Schemes

- No two jobs that require the same reticle should ever be scheduled simultaneously (assuming one reticle set per product)
- Reticle movement and handling should be minimal
- Steppers should only have planned idle time
- Work should be balanced over all slots on all machines
- Scheduling should be proactive
- Schedules should be feasible
**Simulation Model**

**Control Statements**
- Dynamic Reticle Location
- Layer to Slot Assignments
- Layer Processing Times

**INTLC**
- Loading in Initial Conditions

**Arrival**
- Create Entities
- Assign Attributes
- Check Machine and Reticle Availability
- File or Setup Delay and Call Setup

**Setup**
- Adjust Machine & Reticle Characteristics
- Delay for Processing & Call End of Service

**End of Service**
- Determine Job Status
- Collect Statistics
- Schedule Re-Entrant Arrival or Terminate
- Call Setup if NNQ > 0, otherwise Release Machine

---

*SRC/ISMT Factory Operations Research Center*
Network Flow Model

Capacity = # of jobs waiting for the same reticle
Inv Cost = 1 unit penalty
Frc Cost = 2 unit penalty

Capacity = # of jobs possible to process per shift
Cost = 0/15/30 unit penalty for assigning a reticle to a specific machine/slot

Balance of Flow Constraints are applicable to all nodes in the network

Model is driven by restricted arc capacities for each machine
UB = LB = full schedule

Capacity = ∞
Cost = 0

Capacity = ∞
Cost = -60

SRC/ISMT Factory Operations Research Center
Models in Combination

- In the simulation model, time is suspended every 12 hours.
- The network model is loaded and solved for the current situation.
- Reticle locations are instantaneous updated to optimal arrangement.
- Simulation is resumed.
Preliminary Results

<table>
<thead>
<tr>
<th>Instance</th>
<th>Number of Moves</th>
<th>Thousands</th>
</tr>
</thead>
<tbody>
<tr>
<td>11121</td>
<td>MS MOVES</td>
<td>160.00</td>
</tr>
<tr>
<td>11122</td>
<td>MM MOVES</td>
<td>20.00</td>
</tr>
<tr>
<td>11221</td>
<td>NC MOVES</td>
<td>140.00</td>
</tr>
<tr>
<td>11222</td>
<td>MS MOVES</td>
<td>180.00</td>
</tr>
<tr>
<td>12121</td>
<td>MM MOVES</td>
<td>20.00</td>
</tr>
<tr>
<td>12122</td>
<td>NC MOVES</td>
<td>120.00</td>
</tr>
<tr>
<td>12221</td>
<td>MS MOVES</td>
<td>160.00</td>
</tr>
<tr>
<td>12222</td>
<td>MM MOVES</td>
<td>20.00</td>
</tr>
</tbody>
</table>
Future Work

- Determine under what conditions a second reticle set should be purchased
- Perform a designed experiment to determine the impact of various penalty schemes
- Investigate other scheduling methodologies to improve other parameters
- Integrate reticle management methodology in the BATCS code already under development
- Incorporate Reticle Inspection with more detail
- Layer of the penalties for jobs in the forecast queue and determine significance
- Implement into ASAP development framework
Parallel Machines with unequal ready times and due dates

*University of Illmenau*
Scheduling of Jobs with Unequal Ready Dates on Parallel Machines

- **Problem**
  \[ \text{Pm} | r_j, \text{dedication} | \sum w_j \max \left(0, c_j - d_j \right) \]

- **Performance Measure**
  \[ \text{TWT} := \sum w_j \max \left(0, c_j - d_j \right) = \sum w_j T_j \]

- **Related research:** Akturk, Ozdemir, EJOR 2001,
  \[ 1 | r_j | \sum w_j \max \left(0, c_j - d_j \right) \]

- **Local dominance rule**
Scheduling of Jobs with Unequal Ready Dates on Parallel Machines

Three-Phase-Algorithm
- assign jobs to machines via genetic algorithm (GA)
- sequence the jobs on each single machine
- via ATC rule
- improve the quality of the schedule by using the local dominance rule (LDR)

Phase 1
- job-based chromosome representation
- one-point-crossover, flip-mutator
- C++-framework GaLib (MIT 1999) for implementation
Scheduling of Jobs with Unequal Ready Dates on Parallel Machines

Phase 2
• variants of the ATC-Dispatching Rule

\[ I_{1j}(t) := \frac{w_j}{p_j} \exp\left( -\frac{\max(d_j - p_j - (t - r_j))}{k\bar{p}} \right) \]

\[ I_{2j}(t) := \frac{w_j}{p_j} \exp\left( -\frac{\max(d_j - p_j - t)}{k\bar{p}} \right) \left( 1 - \frac{B \max(0, r_j - t)}{\bar{p}} \right) \]

\[ I_{3j}(t) := \frac{w_j}{p_j} \exp\left( -\frac{\max(d_j - p_j - (t - r_j))}{k\bar{p}} \right) \left( 1 - \frac{B \max(0, r_j - t)}{\bar{p}} \right) \]

Phase 3 (optional)
• LDR (Akturk, Ozdemir (2001))
Scheduling of Jobs with Unequal Ready Dates on Parallel Machines

- GA
- Assign Jobs to Machines
  - Sequencing of the Jobs on a Single Machine
    - Calculate $\sum w_i T_i$
  - Sequencing of the Jobs on a Single Machine
    - Calculate $\sum w_i T_i$
- Evaluation of the Complete Schedule
Scheduling of Jobs with Unequal Ready Dates on Parallel Machines

Computational Results

<table>
<thead>
<tr>
<th>TWT</th>
<th>Number of Jobs</th>
<th>Average Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>(GA + ATC)/(GA+ATC+LDR)</td>
<td>n=30</td>
<td>n=60</td>
</tr>
<tr>
<td>m= 2, ( \alpha = 0.50 )</td>
<td>0.6500</td>
<td>0.6900</td>
</tr>
<tr>
<td></td>
<td>0.5000</td>
<td>0.4300</td>
</tr>
<tr>
<td>m= 2, ( \alpha = 0.75 )</td>
<td>0.6700</td>
<td>0.6900</td>
</tr>
<tr>
<td></td>
<td>0.6100</td>
<td>0.5100</td>
</tr>
<tr>
<td>m= 3, ( \alpha = 0.50 )</td>
<td>0.7200</td>
<td>0.6600</td>
</tr>
<tr>
<td></td>
<td>0.6200</td>
<td>0.4300</td>
</tr>
<tr>
<td>m= 3, ( \alpha = 0.75 )</td>
<td>0.7800</td>
<td>0.6700</td>
</tr>
<tr>
<td></td>
<td>0.7200</td>
<td>0.5600</td>
</tr>
<tr>
<td>m= 5, ( \alpha = 0.50 )</td>
<td>0.8300</td>
<td>0.8000</td>
</tr>
<tr>
<td></td>
<td>0.7800</td>
<td>0.6200</td>
</tr>
<tr>
<td>m= 5, ( \alpha = 0.75 )</td>
<td>0.8700</td>
<td>0.8100</td>
</tr>
<tr>
<td></td>
<td>0.8500</td>
<td>0.7100</td>
</tr>
</tbody>
</table>
Parallel Machines with setups, unequal ready times, and due dates

Amit Gadkari

John Fowler, Michele Pfund
Problem Description

- \( \text{Pm} / r_j, s_{jk} / \sum w_j T_j \)
  - \( \text{Pm} \): mc environment with \( m \) parallel mc
  - \( r_j \): Ready time of job \( j \)
  - \( s_{jk} \): Setup time from job \( j \) to \( k \)
  - \( w_j \): Weight of job \( j \)
  - \( T_j \): Tardiness of job \( j \)

- Scheduling \( n \) jobs with unequal release dates and sequence dependent setup times on \( m \) identical parallel machines so as to minimize the total weighted tardiness
• Apparent Tardiness Cost with Setups and Ready times
• Inserted Idle time in the schedule can reduce total weighted tardiness
• Impact of hot jobs on the total weighted tardiness of the schedule

\[ I_j(t) = \frac{w_j}{p_j} \exp \left( -\max \left( d_j - p_j - \max(r_j,t),0 \right) \right) \exp \left( -\frac{s_{lj}}{k_2} \right) \exp \left( -\frac{\max(r_j - t,0)}{k_3 p} \right) \]
Results

- Percentage Improvement in Total Weighted Tardiness with EDD as Base

<table>
<thead>
<tr>
<th>Parameter</th>
<th>WSPT</th>
<th>ATCS</th>
<th>BATCS</th>
<th>ATCSR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Machine Learning Techniques for determining k parameters in BATCs Algorithm

University Of Illmenau
Machine Learning Techniques for Determining the Look-Ahead Parameter in the BATC Rule

**Problem**
- batch processing in the diffusion area of a wafer fab
- very long processing time
- $P_m|\text{batch, incompatible}| \sum w_i T_i$
- NP-hard problem

**Solution**
- ATC (apparent tardiness cost) dispatching heuristic:

\[ I_{ij}(t) := \frac{w_{ij}}{p_j} \exp\left( - \left( \frac{d_{ij} - p_j - t}{\kappa p} \right)^+ \right) \]

- $k$: look-ahead parameter
Machine Learning Techniques for Determining the Look-Ahead Parameter in the BATC Rule

- research question: how to choose k?
- related literature:
  - Lee, Pinedo (1997): formulas for k value based on characteristics of the problem for Pm|SJ| \( \sum w_j T_j \)
  - Park, Kim, Lee (2000): neural network approach for Pm|SJ| \( \sum w_j T_j \)
- optimal value of k (with respect to TWT) depends on:
  - batch per machine factor
  - tightness of the schedule T
  - due date range R
  \[ \mu := \frac{n}{mB} \]
Machine Learning Techniques for Determining the Look-Ahead Parameter in the BATC Rule

- two different approaches:
  - neural networks (extension of the work of Lee and Pinedo to the batch case)
  - inductive decision trees (new in this situation)

- neural network approach
  - tool NeuralWorks Professional II/Plus
  - disadvantage: training phase requires several minutes
Machine Learning Techniques for Determining the Look-Ahead Parameter in the BATC Rule

• **inductive decision tree approach**
  – divide the range of k, i.e., the interval [0.5,5.0] in equidistant classes
  – structure of a learned **decision tree** is a collection of disjoint hyper cubes
  – partitioning the domains of attributes into intervals based on ID3 algorithm (Quinland 1993)
  – test on a decision node is basically a rule
  – over thousand rules

• decision trees: slightly better results compared to neural networks

• less effort in the training phase of decision trees compared to neural networks
Machine Learning Techniques for Determining the Look-Ahead Parameter in the BATC Rule

Factor Combinations:

\[ \mu = (10, 15, 20, \ldots, 60) \]
\[ T = (0.30, 0.35, 0.40, \ldots, 0.90) \]
\[ R = (0.50, 0.75, 1.00, \ldots, 5.00) \]

Mean Square Error for Neural Network Approach

<table>
<thead>
<tr>
<th>Nodes of the Hidden Layer</th>
<th>Error Training Data</th>
<th>Error Test Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0.0680</td>
<td>0.0829</td>
</tr>
<tr>
<td>7</td>
<td>0.0535</td>
<td>0.0717</td>
</tr>
</tbody>
</table>
## Machine Learning Techniques for Determining the Look-Ahead Parameter in the BATC Rule

### Mean Square Error for Inductive Decision Tree Approach

<table>
<thead>
<tr>
<th>Number of Considered Classes</th>
<th>Error Training Data</th>
<th>Error Test Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.0641</td>
<td>0.1020</td>
</tr>
<tr>
<td>10</td>
<td>0.0450</td>
<td>0.0940</td>
</tr>
<tr>
<td>15</td>
<td>0.0415</td>
<td>0.0941</td>
</tr>
<tr>
<td>20</td>
<td>0.0396</td>
<td>0.0797</td>
</tr>
<tr>
<td>25</td>
<td>0.0399</td>
<td>0.0914</td>
</tr>
</tbody>
</table>

### Mean Square Error for Inductive Decision Tree Approach (Optimized Tree)

<table>
<thead>
<tr>
<th>Number of Considered Classes</th>
<th>Error Training Data</th>
<th>Error Test Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.0645</td>
<td>0.1020</td>
</tr>
<tr>
<td>10</td>
<td>0.0422</td>
<td>0.0807</td>
</tr>
<tr>
<td>15</td>
<td>0.0383</td>
<td>0.0784</td>
</tr>
<tr>
<td>20</td>
<td>0.0354</td>
<td>0.0679</td>
</tr>
<tr>
<td>25</td>
<td>0.0360</td>
<td>0.0752</td>
</tr>
</tbody>
</table>
Machine Learning Techniques for Determining the Look-Ahead Parameter in the BATC Rule

IDT (20 Classes), Test Data

Look Ahead Parameter $k$

Measured  Estimated  Difference

SRC/ISMT Factory Operations Research Center
Parallel Batch Machines

Hari Balasubramanian

Lars Moench, John Fowler, Michele Pfund, Matt Carlyle
880.002: Subproblem Solution Procedures

- **Parallel Batch Machine Scheduling**

- \( Pm | B, \text{incompatible} | \sum w_j T_j \)
Parallel Batch Machines: GA Version 1

Form Batches

GA

Assignment of batches to machines

Sequence of batches in machine 1 → Apply Sequencing Heuristic for single machine → Calculate $\sum w_j T_j$

Sequence of batches on machine M → Apply Sequencing Heuristic for single machine → Calculate $\sum w_j T_j$

Evaluation of complete schedule
Parallel Batch Machines: GA Version 2

- GA
- Assignment of jobs to machines
  - Form Batches on Machine 1
  - Form Batches on Machine M
  - Apply Sequencing Heuristic for single machine
  - Calculate $\sum w_j T_j$
  - Apply Sequencing Heuristic for single machine
  - Calculate $\sum w_j T_j$
  - ... Calculate $\sum w_j T_j$
- Evaluation of complete schedule
## Parallel Machines with Batching GA: Results

<table>
<thead>
<tr>
<th></th>
<th>EDD</th>
<th>ATC</th>
<th>ATC</th>
<th>ATC</th>
<th>ATC</th>
<th>ATC</th>
<th>ATC</th>
<th>GA2</th>
<th>GA2</th>
<th>GA2</th>
<th>GA2</th>
<th>GA2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compare</td>
<td>ATC</td>
<td>ATC</td>
<td>GA1a</td>
<td>GA1a</td>
<td>GA1a</td>
<td>GA1a</td>
<td>GA1b</td>
<td>GA1b</td>
<td>EDD</td>
<td>ATC</td>
<td>ATC</td>
<td>ATC</td>
</tr>
<tr>
<td>ATC Batc</td>
<td>BATC</td>
<td>BATC</td>
<td>EDD</td>
<td>BATC</td>
<td>DH</td>
<td>BATC</td>
<td>DH</td>
<td>EDD</td>
<td>BATC</td>
<td>DH</td>
<td>BATC</td>
<td>DH</td>
</tr>
<tr>
<td>swap</td>
<td></td>
<td></td>
<td>swap</td>
<td>swap</td>
<td>swap</td>
<td>swap</td>
<td></td>
<td></td>
<td>swap</td>
<td>swap</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Batch Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jobs/family</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Due-Date Variance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>% of Tardy Jobs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>1</td>
<td>0.96</td>
<td>0.91</td>
<td>0.92</td>
<td>0.91</td>
<td>1.74</td>
<td>1.27</td>
<td>1.26</td>
<td>0.96</td>
<td>0.96</td>
<td>1.19</td>
<td>1.15</td>
</tr>
<tr>
<td>m=3</td>
<td>1</td>
<td>0.98</td>
<td>0.99</td>
<td>0.98</td>
<td>0.92</td>
<td>0.91</td>
<td>1.69</td>
<td>1.25</td>
<td>1.23</td>
<td>0.94</td>
<td>0.94</td>
<td>0.92</td>
</tr>
<tr>
<td>Batch Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jobs/family</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>1</td>
<td>0.96</td>
<td>0.91</td>
<td>0.92</td>
<td>0.91</td>
<td>1.74</td>
<td>1.27</td>
<td>1.26</td>
<td>0.96</td>
<td>0.96</td>
<td>1.19</td>
<td>1.15</td>
</tr>
<tr>
<td>m=4</td>
<td>1</td>
<td>0.96</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
<td>0.91</td>
<td>0.91</td>
<td>1.69</td>
<td>1.25</td>
<td>1.23</td>
<td>0.94</td>
<td>0.94</td>
</tr>
<tr>
<td>Batch Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jobs/family</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>1</td>
<td>0.96</td>
<td>0.91</td>
<td>0.92</td>
<td>0.91</td>
<td>1.74</td>
<td>1.27</td>
<td>1.26</td>
<td>0.96</td>
<td>0.96</td>
<td>1.19</td>
<td>1.15</td>
</tr>
<tr>
<td>m=5</td>
<td>1</td>
<td>0.96</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
<td>0.91</td>
<td>0.91</td>
<td>1.69</td>
<td>1.25</td>
<td>1.23</td>
<td>0.94</td>
<td>0.94</td>
</tr>
<tr>
<td>Batch Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jobs/family</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>1</td>
<td>0.96</td>
<td>0.91</td>
<td>0.92</td>
<td>0.91</td>
<td>1.74</td>
<td>1.27</td>
<td>1.26</td>
<td>0.96</td>
<td>0.96</td>
<td>1.19</td>
<td>1.15</td>
</tr>
</tbody>
</table>
Development of Data Structure and Integration

Lars Moench, Detlef Pabst, Michele Pfund

Hari Balasubramanian, Matt Carlyle, John Fowler, Oliver Rose
Development of a data structure for SSPs

- The Task 2 team along with the Task 4 team developed a data structure for integrating SSPs into the ASAP-Shifting Bottleneck environment.

- The basic framework consists of an abstract base class (called WIS_SSP_Basic) from which a number of SSP classes (representing rules such as EDD, ATCS, BATCS, Batch_EDD, WSPT etc.) are derived.

- The structure allows derived classes to have features unique to them while inheriting the main features of the base class.
Development of a data structure for SSPs

Abstract Base Class
WIS_SSP
{
    Virtual solve_problem()
}

Derived Class
EDD_SSP
{
    Solve_problem()
}

Derived Class
ATCS_SSP
{
    Solve_problem()
}

..............

Derived Class
Simple_Reticle_SSP
{
    Solve_problem()
}
Development of a data structure for SSPs

• The data structure was developed during the summer months, implemented in August, and validated in September.

• Based this structure, a validated library of SSP rules has been developed this year and currently consists of:
  – ATCS, BATCS
  – EDD, Batch EDD
  – FIFO, Batch FIFO
  – Least Slack
  – Critical Ratio
  – SPT and WSPT
  – Highest Priority
  – Reticle Management* (in progress)
Testing of SSPs

- The SSPs developed were then tested on the simple minifab model with three workstations
  - Station 1 (consisting of batching machines A and B)
  - Station 2 (consisting of 2 machines C and D)
  - Station 3 (consisting of a single machine E)

- At each of these stations different SSPs could be used

- Each station is considered a factor, its levels being the different SSPs that could be used at the station

- For the batch station (Station 1) 3 different SSPs can be used: BATCS, Batch FIFO, and Batch EDD

- For the non-batch stations (Stations 2 and 3) 8 different SSPs can be used: ATCS, EDD, FIFO, WSPT, SPT, Critical Ratio, Least Slack, Highest Priority
Testing of SSPs (cont.)

- Thus a designed experiment with 3 x 8 x 8 runs was conducted on the minifab model

<table>
<thead>
<tr>
<th>Station 1</th>
<th>Station 2</th>
<th>Station 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch EDD</td>
<td>ATCS</td>
<td>ATCS</td>
</tr>
<tr>
<td>Batch FIFO</td>
<td>EDD</td>
<td>EDD</td>
</tr>
<tr>
<td>BATCS</td>
<td>FIFO</td>
<td>FIFO</td>
</tr>
<tr>
<td>WSPT</td>
<td>WSPT</td>
<td>WSPT</td>
</tr>
<tr>
<td>SPT</td>
<td>SPT</td>
<td>SPT</td>
</tr>
<tr>
<td>Critical Ratio</td>
<td>Critical Ratio</td>
<td>Least Slack</td>
</tr>
<tr>
<td>Least Slack</td>
<td>Least Slack</td>
<td>Highest Priority</td>
</tr>
<tr>
<td>Highest Priority</td>
<td>Highest Priority</td>
<td></td>
</tr>
</tbody>
</table>
Results: Best Configuration

<table>
<thead>
<tr>
<th>Station 1</th>
<th>Station 2</th>
<th>Station 3</th>
<th>TWT</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch_ATCS</td>
<td>EDD</td>
<td>EDD</td>
<td>202.917</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>FIFO</td>
<td>EDD</td>
<td>202.917</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>LS</td>
<td>EDD</td>
<td>202.917</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>CR</td>
<td>EDD</td>
<td>202.917</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>ATCS</td>
<td>EDD</td>
<td>202.917</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>EDD</td>
<td>FIFO</td>
<td>203.059</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>ATCS</td>
<td>FIFO</td>
<td>203.059</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>FIFO</td>
<td>FIFO</td>
<td>203.059</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>CR</td>
<td>FIFO</td>
<td>203.059</td>
<td>53</td>
</tr>
<tr>
<td>Batch_ATCS</td>
<td>WSPT</td>
<td>EDD</td>
<td>206.942</td>
<td>53</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pure Dispatching</th>
<th>TWT</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO</td>
<td>330.072</td>
<td>53</td>
</tr>
<tr>
<td>EDD</td>
<td>274.738</td>
<td>53</td>
</tr>
<tr>
<td>CR</td>
<td>266.542</td>
<td>53</td>
</tr>
<tr>
<td>LS</td>
<td>274.738</td>
<td>53</td>
</tr>
</tbody>
</table>

Improvements:
38% over FIFO
26% over EDD & LS
24% over CR
Current and planned SSP's

• Current implementation:
  – Batch Machines: BATCS, Batch_EDD, Batch FIFO
  – ATCS, EDD, FIFO, Least Slack, Critical Ratio, SPT, WSPT, Highest Priority, Reticle Management*

• Planned implementation:
  – All methodologies discussed today!
  – Incorporation of optimization libraries to enable GA, SA, TS, LP, and IP based solution methodologies
  – Hierarchical SSP solution methodology structure

• Liaison Suggestions????
880.002: Subproblem Solution Procedures

- **Future Plans and Directions**
  - Fully Test Subproblem Solution Procedures Developed in Year 1-2
  - Develop new Subproblem Solution Procedures
    - Tool group dependent SSP’s (including tool dedication)
    - Reticle Management / Generic Resource Dedication
  - Investigate Alternative Performance Measures
  - Investigate Multiple Performance Measures
  - Investigate Hierarchical Method of SSP Selection
  - Integrate PM orders as “scheduled job” (UM / UC integration)
880.003: Statistical Operations Control

Research Personnel:

- Task Leader: George C. Runger, ASU
  Esma Gel, ASU

- Faculty:
  - Michele Pfund, ASU
  - Matt Carlyle, NPS
  - Steven Brown, ASU

- Students
  - Andy Burhanuddin, ASU, MS 6/02 (defended his thesis in 6/02)
  - Cem Vardar, ASU, PhD 8/04

- Liaisons
  - Mani Janakiram, Intel
Primary Anticipated Results:

• Develop decision rules that trigger rescheduling at an opportunistic time

• Create an adaptive rescheduling strategy that considers:
  – Difference between the actual vs. predicted schedule performance
  – Potential improvements to be gained from rescheduling/recovery
880.003: Statistical Operations Control

Research Description:

- **Year 1** – Identify useful metrics from operations to be included in decision rules. Develop approaches for statistical operations control (SOC) that detects significant variations (of multiple variables) from the schedule so that corrective action may be taken.

- **Year 2** – Generate appropriate distance metrics to characterize departures. Consider potential cost benefits from rescheduling and use multivariate statistical process control techniques and simulation.

- **Year 3** – Enhance and modify the trigger rules to enable their integration with rescheduling algorithms.
880.003: Statistical Operations Control

**Deliverables:**

- Report on development of triggers for adaptive rescheduling. Report (Year 1)

- Report on comparisons of adaptive rescheduling with alternative methods. Report (Year 2)

Executive Summary:

The last year has been focused upon evaluating possible event and data triggers that might be used to decide whether it might be desirable to generate a new schedule or take some other corrective action.
880.003: Statistical Operations Control

- **Year 1 Accomplishments and Progress**
  - Identification of event triggers that may initiate rescheduling
    - preliminary list complete, additions continue
  - Identification of data triggers that may initiate rescheduling:
    - several metrics evaluated, more to consider
  - Development of algorithms to differentiate between normal variability and status changes that require an action
  - Determine taxonomy of triggering mechanisms
  - Exploration of appropriate monitoring strategies
    - time intervals, event based
880.003: Statistical Operations Control

- **Year 2 Accomplishments and Progress**
  - Andy Burhanuddin completed his MS thesis. He evaluated different data triggers to detect out-of-control behavior in the system.
  - Evaluate a trigger: impact to schedule versus fab nervousness
    - current plan is to evaluate through disjunctive graph
    - tradeoff computational burden with schedule performance
    - tradeoff next planned scheduling with rescheduling immediately
  - Team got the expertise on how the software works and how to make additions for Task 3’s coding needs
  - A library structure has been developed for rescheduling triggers (thanks to Detlef Pabst and Dr. Pfund). Most promising triggers (according to results of Burhanuddin’s MS thesis) have been added to this library
  - Initial experimentation has been started with the software. First step is to find out under which conditions rescheduling is beneficial.
Triggers for Rescheduling

**Event-based**
Change in *system status* in terms of:
- Machines
- Operators
- Job priorities/specs
- New jobs arrive
- Etc.

**Data-based/Statistical**
*Deviations* due to various sources of variability:
- Change from scheduled completion time
  - Make use of internal due dates throughout process
- Number and type of jobs in critical queues
- Fraction of rework/yield
- Etc.

**Key:** Finding effective triggers that will minimize computational burden and unnecessary rescheduling.

⇒ Ability to *discriminate* “normal noise” from “trends” that call for action (i.e., either repair or reschedule)
⇒ Impact evaluation critical: schedule performance versus excessive reschedules
⇒ Time dependency of actions and other open issues...
Example Trigger Mechanism

1. **Machine Fails**
2. **A Job is Completed at a Station**
   - If $C_{ij} > d_{ij} + \varepsilon$?
     - **Disjunctive Graph Update**
     - **Reschedule?**
       - **YES** Generate New Schedule
       - **NO** Continue Monitoring
3. **Reschedule?**
   - **YES** Generate New Schedule
   - **NO** Continue Monitoring
Event-based Triggers - Examples

- Machine break-downs/completion of repairs
- New job arrives
- Preventive maintenance operations
- Operator unavailability/absenteeism
- Material handling system delays/break-downs
- Changes in jobs: priorities/ specifications/ orders, etc.

Data-based/Statistical Triggers - Examples

- Tracking various variables such as
  - Deviations from internal due dates throughout the fab
  - Actual versus predicted deviation from revised disjunctive graph
  - Number and type of jobs in the critical queues
  - Number of consecutive jobs deviating from the estimated schedule
  - Fraction of scrap/ rework, etc.
**SOC Approach** Data-based/Statistical Triggers

- Schedule described by a disjunctive graph/matrix of jobs and fab steps with internal due dates
- Trigger can compare an initial schedule with a real-time update
- Unifying approach that links triggers to SBH and repairs
  - Compare matrices that include actual and **predicted** completion times for current and initial schedules
  - Predicted times expected to provide more proactive response
  - Measures slack consumed at completed steps and predicts for future steps
  - Question framed as matrix/graph metrics/evaluation
  - Frequency of updates a tradeoff between computational complexity and schedule trigger performance
  - Dispatching rules incorporated into triggers through matrix/graph
- Impact of event or data trigger can be evaluated through future completion times
  - Graduated response:
    - Update disjunctive graph/matrix based on completion time
    - Reschedule based on impact from graph
    - Leads to a direct decision rule: Signal when predicted slack is negative
Main Results of Burhanuddin’s thesis:

- Extensive testing has been carried out by simulations in SLAM and following SOC triggers have been analyzed
  - Deviation in waiting time of lots in queue at every step
  - Deviation in WIP over time
  - Deviation in cumulative WIP at every step over time
  - Deviation in completion times of lots at every step
- Key observations
  - Deviation in Queue Time, Cumulative WIP is not effective as detecting shifts in the process
  - Deviation in WIP can detect the shifts only after a long time.
  - Deviation in Completion time performs best but it is prone to false alarms
- Statistical Operations Control can be applied by using:
  - High run sum statistic sums the positive deviation of the observations in a run
  - High run length statistic counts the length of observations in a run above the planned schedule
  - Control limits are estimated using extreme percentiles empirically [Runger et al., 1998]
## 880.003: Statistical Operations Control

### Actual, real-time, and predicted performance

<table>
<thead>
<tr>
<th>Job/Step</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Actual completion time</td>
<td>Actual completion time</td>
<td><em>Calculated</em> completion time</td>
<td><em>Calculated</em> completion time</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Scheduled performance

<table>
<thead>
<tr>
<th>Job/Step</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><em>Scheduled</em> completion time</td>
<td><em>Scheduled</em> completion time</td>
<td><em>Scheduled</em> completion time</td>
<td><em>Scheduled</em> completion time</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
880.003: Statistical Operations Control

- The deviation between predicted and scheduled completion times can be used in two ways
  - As a data trigger (calculated periodically)
  - As a method to evaluate the impact of an event trigger (calculated when an event trigger occurs)
- Predicted completion times calculated via deterministic simulation=
- Two ways to calculate Predicted Completion times
  - Use a separate simulation model
    - Flexible
    - Difficult to maintain and integrate
  - Use disjunctive graph
    - Not flexible
    - Relatively easier to maintain and integrate
- Code to calculate predicted completion times is in progress
880.003: Statistical Operations Control

- 5 replications of MINIFAB model using FIFO schedule
- 40, 80 & 160 minutes breakdowns are inserted at t=3500 at bottleneck
880.003: Initial Experimentation Results

- Initial experiments were run in MINIFAB with inserted breakdowns
- Run length is 30 days and the time between breakdowns were 5 days (deterministic), 325 jobs.
- Breakdowns of 160, 300, and 500 minutes were inserted
- Three cases:
  - FIFO dispatching
  - Schedule every 12 hours
  - Schedule every 12 hours with reschedule after breakdown
- Rescheduling was done after the machine comes up
- Promising results were obtained (essentially equivalent outs)

### Total Weighted Tardiness with Inserted breakdowns on Batching Station in MIINIFAB

<table>
<thead>
<tr>
<th>BD Duration(min)</th>
<th>FIFO</th>
<th>Shifting Bottleneck</th>
<th>Shifting Bottleneck with Rescheduling</th>
<th>Overall Improvement with rescheduling</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>1430.01</td>
<td>340.61</td>
<td>339.21</td>
<td>76.28%</td>
</tr>
<tr>
<td>300</td>
<td>2629.94</td>
<td>528.83</td>
<td>469.12</td>
<td>82.16%</td>
</tr>
<tr>
<td>500</td>
<td>4051.50</td>
<td>1367.05</td>
<td>1112.61</td>
<td>72.54%</td>
</tr>
</tbody>
</table>
Future Plans and Directions

- Year 3’s goal is to enhance and modify the trigger rules to enable their integration with rescheduling algorithms and test the performance of rescheduling triggers.
- Also, continue work on unifying approach:
  - Schedule evaluated through disjunctive graph/matrix of jobs and fab steps with internal due dates.
  - Evaluate frequency of updates versus performance.
  - Evaluate efficacy of future completion times for data triggers.
- Link triggers to SBH and repairs.
- Possible experimentation factors and issues:
  - Scheduling horizon while scheduling/rescheduling.
  - Determining the rescheduling threshold(s).
  - How frequently should we call the rescheduling routine?
  - Effect (numerical measure) of rescheduling.
  - Reschedule when the machine first breaks down or reschedule after the breakdown or both?
  - Utilization level?/Due date tightness?
  - Global vs local thresholds?
880.004: Testing, Comparison, and Implementation Issues

RESEARCH PERSONNEL:

• Task Leader: Oliver Rose, UW
• Faculty:
  – Lars Mönch, TUI
  – Roland Sturm, IPA
• Students
  – Detlef Pabst, UW, MS 6/02
  – Jens Zimmermann, TUI, MS 12/02
  – Florian Holzinger, UW, MS 6/03
  – Kai Hübner, TUI, MS 6/03
  – Torsten Schiele, UW, MS 6/03
  – Maryam Zehtaban, IPA, PhD TBD
• Liaisons
  – Mathias Schulz, Infineon
  – Sidal Bilgin, LSI
  – Joanna Shear, Intel
• Additional Support
  – Joey Skinner, Brooks/ASI
Primary Anticipated Results:

- Develop a testing and development framework
  - simulation package linked to implementations of our methodologies

- Implement our methodologies
  - scheduling,
  - statistical operational control,
  - and rescheduling methodologies.

- Documentation for integration of these methods with MES systems.
880.004: Testing, Comparison, and Implementation Issues

**Research Description:**

- **Year 1** - Investigate simulation and optimization packages, and combine to provide the functionality we need. Implement various dispatching rules and test them in our environment.

- **Year 2** - Implement our scheduling approaches, including the special SSPs for various equipment types. Compare statistically as well as on an instance-by-instance basis with dispatching rules. Start implementing and testing SOC procedures.

- **Year 3** - Further integration of various SSPs, reticle management approaches, SOC signaling mechanism(s), and the rescheduling methodology into the decision support system. Test on long time-horizon simulation runs that model various situations that may arise in the wafer fab context. Document how the various components of our methodology can be integrated with an MES system to provide real-time operational support in wafer fabs.
880.004: Testing, Comparison, and Implementation Issues

**Deliverables:**

- Analysis of available software and choice for environment development. Report (Year 1)
- Environment development, with implementation and testing of basic dispatching rules using MASM Lab Testbed datasets. Software (Year 1)
- Report on implementation and performance of scheduling approach, with various SSPs, vs. dispatching rules. Report (Year 2)
- Report on implementation and testing of SOC methodology. Report (Year 2)
- Implementation of full decision support system in our testing environment, including advanced SSPs, sophisticated rule sets for SOC, and intelligent rescheduling. Software (Year 3)
- Report on implementation and integration of our methods with MES systems, including guidelines for maintenance and future development. Final Report (Year 3)
880.004: Testing, Comparison, and Implementation Issues

- **Executive Summary:**
  - Developed data model as interface between fab and scheduler: data model is going to be used by all 4 tasks (common link to the test environment)
  - Customized ASAP model of MiniFab Model, MIMAC Test Bed Data Set 1 Model, and ISMT 300mm Fab Model to export/import data to/from scheduler (via data model)
  - SSP Framework development (generic class for Task 2)
  - Speed and quality improvement for SBH
  - Full integration of object-oriented database
  - Pilot runs for SSP and Scheduler parameter setting
  - Experimental environment GUI
880.004: Testing, Comparison, and Implementation Issues

- **Accomplishments and Progress**
  
  - Review of several simulation packages to assess applicability in the project:
    
    - Brooks/ASI ASAP selected
      - Used in semiconductor industry
      - Flexibility: can be customized with C++
880.004: Testing, Comparison, and Implementation Issues

• **Accomplishments and Progress (contd.)**

  • Development of an ASAP customization concept
    • ASAP is used for fab emulation
    • Using the Scheduler as a stand-alone tool must be possible
  
  • Development of a data model as an interface
    • For all tasks
    • For all implemented modules, e.g. simulation model and scheduler
  
  • Permanent improvement of the data model to serve the requirements of all tasks
880.004: Testing, Comparison, and Implementation Issues

Structure of the Simulation & Testing Environment

- ASAP Simulation Model
- The Real Fab (MES, ERP,...)
- Bus
- Scheduler
- Data Model
- OO DB

Dispatcher
880.004: Testing, Comparison, and Implementation

Issues

• Accomplishments and Progress (contd.)

• Implementation of customization and data model:
  • Lot-list-based dispatch rules for ASAP (both for single lot and batch equipment)
  • Initialization of the data model from ASAP model text files
  • Implementation of the fab messages functions to update the data model
  • Implementation scheduler call and delayed reception of the scheduler results
  • Integration of the above parts
**880.004: Testing, Comparison, and Implementation Issues**

- **Accomplishments and Progress (contd.)**
  - Replacement of Rogue Wave Lib Classes by STL classes
  - Implementation of FIFO, EDD, ATCS, BATCS, and CR schedulers to test customization
  - Interfacing to Scheduler
  - Implementation of functions to record historical information about all lots
  - Implementation of a Lot-Viewer GUI for the data model
  - Fab models used:
    - MiniFab Model
    - MIMAC Test Bed Data Set 1
    - ISMT 300mm Fab Model (without AMHS)
880.004: Testing, Comparison, and Implementation Issues

- **Accomplishments and Progress (contd.)**
  - Source code improvement / Re-implementation of code
  - Speed improvements of the SBH algorithm
    - Testing various memory allocation procedures
    - Testing different approaches for graph implementation, e.g. topological sorting of nodes
    - Efficient data structures for fast object access
  - Quality improvement of Scheduler
    - Adaptation of BATCS rule to dynamic environment
    - Cycle avoidance procedures for all SSPs
  - Full integration of POET Fast Objects database (OO DB)
    - User-specific backups of the content of the data layer
    - Initialization of data layer and fab graph after failures
880.004: Testing, Comparison, and Implementation Issues

Accomplishments and Progress (contd.)

- Development of a GUI for experimentation environment
- Pilot runs
  - Parameter setting of core SBH, e.g., scheduler horizon
  - Parameter setting of individual SSPs, e.g., BATCS
  - Approaches to determine operation due dates
880.004: Testing, Comparison, and Implementation Issues

• Organizational Environment

  • CVS (concurrent version system) server as source code repository with revision control (for all tasks)

  • Implementation of a hierarchical meeting structure
    • Weekly local meetings at Würzburg, Ilmenau, and Stuttgart
    • Bimonthly meetings of the German group (researchers and students)
    • Phone meetings on demand
Framework for the Performance Assessment of the Shifting Bottleneck Heuristic for Complex Job Shops
Interfacing of Shop-floor Control and Fab

• Future IT architecture for wafer fabs must be open and comply with (de facto) standards such as Common Object Request Broker Architecture (CORBA) specifications.

• After the performance assessment only the functions interfacing between simulation and scheduler have to be replaced by functions interfacing to factory IT systems, e.g., MES or ERP.

• Due to the architecture no further changes in the control software are required.

• To obtain an appropriate interface with little extra work the following interfaces have to be provided (SEMI E-105-0701):
  – *Scheduling Service Interface*: communicate scheduling information to the fab
  – *Scheduling Factory Input Interface*: communicate fab information to the scheduling component
Framework Architecture

ASAP Simulation Model

Dispatcher

The Real Fab (MES, ERP,...)

Dispatcher

CORBA Message Bus

Scheduler

Data Model
Simulation Environment

- Main purpose is to emulate the wafer fab
- Simulation software has to provide appropriate functionality to model wafer fabs
- Simulation software has to be able to communicate with an external shop-floor control component:
  - Time-driven: scheduler triggered by a timer
  - Event-driven: shop-floor triggers scheduler

```
ASAP  <->  Data Model  <->  Scheduler
     Event driven         Time driven
```
Data Model Concept

Data Model usage:
• Initialization at the beginning of the simulation
• Update of objects during the simulation run in an event-driven manner
• Shop-floor control component reads information from data model

Data Model update:
• Data model update is event-driven
• Data transfer from simulation to the data model should take place when interested events happen
• These events should be mapped to ASAP internal events

Notification/subscription/publication mechanism in ASAP is the approach to gain these functionality
The Publish/Subscribe Notification System

• AP’s publish/subcribe notification system is a messaging system that allows FIEntities to inform your customization when certain events occur.

• When an event occurs that an FIEntity wants to publicize, it publishes a notification.

• In the simulation model extension, you subscribe to notifications by writing notification functions, and then adding subscriptions for the functions to the publishing entities.

• “Publishing” means that a FIEntity calls all the notification functions for which it has subscriptions for a given event.
Overview of Fraunhofer Activities

- Currently: Testing Scheduler with a large Fab Model
- Investigation on:
  - Factory ramp-up
  - Requirement for additional events for better tracking scheduling process
  - Test scheduler in factory ramp-up process
  - Change of static data during simulation
Run Time Improvement of the FORCe Scheduler

Andre Pfeuffer
Motivation

- Scheduling of the MIMAC test bed data set 1 took more than one hour per simulation day.
- Profiling analysis showed that the calculation of due dates and ready times requires more than 80% of the time calculating the schedule.
- Further code analysis has shown that implementation of the code was done using a modification of Dijkstra’s shortest path algorithm to calculate the longest path.
Introduction

- The ready times and due dates are needed as input of the subproblem solution procedure (SSP)
- Without re-optimization the number of calls to the ready time algorithm is $2n$, where $n$ is the number of tool groups
- However simulation studies have shown that re-optimization plays an important role in scheduling to improve factory performance
- To measure the benefit of re-optimization the calculation is required two times. Once before the critical toolgroup and once after it.
Implementation using DFS

- Implementation of a topological sort of the nodes of the graph
- Suppose the following sample graph representation

Calculation direction of the ready times

Seq := S,(2,2),(1,2),(1,1),(2,1),(3,1),(4,2),(3,2),(1,3),(2,3),(4,3),T
Implementation using DFS

▷ Due dates were calculated from node T (Cmax)
   (= inverted topological order )

Calculation direction of the due dates
Scheduling in the Dynamic Environment of an ASAP Simulation

Detlef Pabst
Overview

Motivation:

Providing a dynamic testing framework for the shifting bottleneck algorithm using the ASAP simulation environment

My Tasks:

• Customization of ASAP for external scheduling
• Modification of the data model and the scheduler to fit to the dynamic setting
• Validation and debugging of the scheduler
• Development and implementation of a cycle avoidance for the current SSP’s
Customization of ASAP to use external Scheduling

- **FIFactory**
- **FIRes**
- **WIS_Factory** (Data Model)
- **PM Lot**
- **Scheduler**
- **WIS_Schedule**

**Model-Files**
- initialize
- update

**FIFactory**
- ASAP
- initialize
- start periodically, and wait for
- ASAP
- activate
- create and delay

**FIRes**
- ASAP
- used for task selection
- ASAP user attribute

**WIS_Schedule**
- ASAP
- launch
- update

**WIS_Factory**
- read dynamic data
- read static data

**PM Lot**
- ① launch
- ② activate
- ③ create and delay

**Update event**
- launch

**ASAP user attribute**
- rule_WIS_Schedule
Application of the scheduler results in the simulation:

- WIS_Schedule == list of strings like:
  
  $12870 \& L_{Pa_1}@S1 : L_{Pb_1}@S1:
  
  ready time $\\quad$ lot name $\\quad$ step name

- rule_WIS_Schedule:
  
  - take first entry from WIS_Schedule
    
    $\rightarrow$ try to find the corresponding lots in the station family worklist
  
  - all lots found $\rightarrow$ start the batch
  
  - batch not complete & ready time is not yet reached
    
    $\rightarrow$ delay task selection until ready time
  
  - batch not complete & ready time has already passed by
    
    $\rightarrow$ start incomplete batch or try next one
Customization of ASAP

- The scheduler obtains all information from the Data Model
  \[\rightarrow\] real fab: just connect the Data Model to the MES
- The delayed update of the WIS_Schedule objects imitates the progress in the fab state during the schedule calculation.
- Used Model $\rightarrow$ MIMAC testbed data set 1
- Model adaptations for the scheduler ASAP extension:
  - init function
  - PM Order with new PM action
  - task selection rule
  - user attributes to entities FACTORY and STATION
- Testing $\rightarrow$ dummy scheduler implementing FIFO, EDD and CR.
Data Model Test Simulations

Torsten Schiele
Overview

Motivation: Creating a dummy scheduler using dispatching rules (FIFO, EDD, CR) to test the data model and the behavior of our scheduling extension of ASAP.

My tasks:
- Implementation and modification of the FIFO, EDD, CR dispatching rules
- Simulating, testing and comparing their behavior against the dispatching rules of ASAP
Experiments

- After the implementation of the dispatching rules for the small factory model, modifications were necessary to work with the large model.
- Simulation experiments using the large model with ASAP dispatching rules and external WIS dispatching rules.
- If there are differences between the cycle times of the lots:
  - The data model is not correct and/or outdated.
  - The scheduling extension (or its link to ASAP) has some bugs.
FIFO results

Average Cycle Time in days vs Load in %

- FIFO_WIS
- FIFO_ASAP
EDD results

Graph showing the relationship between load in % and AvgCycleTime in days for EDD_WIS and EDD_ASAP.
CR results

![Graph showing CR results]

- CR_WIS
- CR_ASAP

Average Cycle Time in days vs. Load in %
Conclusion

- The results of the external dispatching with FIFO and EDD almost match the ASAP results.
- Thus extension link and data model work fine.
- The large differences between external CR and ASAP CR indicate that there is a problem in the CR implementation. Fixing is in progress.
880.004: Testing, Comparison, and Implementation Issues

- **Future Plans and Directions**
  - Further speed improvement of SBH and SSP code
  - Full integration of Scheduler into ISMT 300mm Fab Model
  - Parameter setting studies
  - Replacing of the current graph algorithms by functions from a professional library (LEDAC)
Technology Transfer:  
Year 1 Project Deliverable Reports

- Task 1 Report on development and implementation of the shifting bottleneck approach and the basic rescheduling methodology.

- Task 1 Report on “State of the Art Scheduling Survey Results: An analysis of a survey sent to 14 SRC and INTERNATIONAL SEMATECH member company wafer fabrication facilities throughout Europe, Asia, and North America.”

- Task 2 Report on reticle management models implementation.


- Task 4 Report on analysis of available software and choice for environment development.
Technology Transfer: Year 2 Project Deliverable Reports

- Task 1 Report on methods to combine tool groups into non-critical areas and abstract them out of the models
- Task 1 Report on the incorporation of material handling issues into shifting bottleneck approach
- Task 2 Report on cluster tools model implementation
- Task 3 Report on comparisons of adaptive rescheduling with alternative methods
- Task 4 Report on implementation and performance of scheduling approach, with various SSPs, vs. dispatching rules.
- Task 4 Report on implementation and testing of SOC methodology
Technology Transfer: Publications


Technology Transfer: Publications


Technology Transfer: Presentations

- Two sessions at INFORMS Annual Meeting, San Jose, CA, Nov.17-20, 2002.


- Session at Flexible Automation and Intelligent Manufacturing (FAIM) Conference,, Dresden, Germany, July 15-17, 2002.

Technology Transfer: Software Developed

- Reoptimization
- SSP Library (SSP’s specified by station within ASAP) - FIFO, EDD, ATCS, Batch-FIFO, Batch-EDD, BATCs, SPT, WSPT, Critical Ratio
- Machine Criticality Measures – Total Weighted Tardiness of Schedule or Pinedo
- Cycle Avoidance
- Time Accuracy window in which to determine whether or not to follow schedule
- Flexible rescheduling time window – Determines time of next schedule implementation (i.e. shiftly)
- Flexible Time Horizon Window to determine how far ahead to schedule jobs in system (i.e. generate schedule for next shift hours based on anticipated X=24 hours of potential processing)
Technology Transfer: Software Developed

- Rescheduling Library
  - Rescheduling by workstation (triggers specified by station within ASAP)
    - Actual minus Expected Completion Time
  - Rescheduling by order (triggers specified by station within ASAP)
    - Actual minus Expected Completion Time
  - Rescheduling by event
    - Equipment Breakdowns
- Delayed Schedule Implementation – i.e. collect information X=2 hours before end of shift and generate (but not implement) schedule, then at shift change time implement schedule
- Interim Due Date Estimation – Simulation, Linear, Exponential