# Supervised Learning Linear Priority Dispatch Rules for Job-Shop Scheduling

Helga Ingimundardottir & Thomas Philip Runarsson

School of Engineering and Natural Sciences, University of Iceland

20th of January, 2011

# Overview

# Goal

- General goal is how to search for *good* solutions for an arbitrary problem domain.
- To automate the design of optimization algorithms.
- In this work we learn new dispatching rules for JSSP
- Using randomly sampled problem instances and their corresponding optimal solutions.

# Previous work

Methods previously proposed for solving JSSP:

- Genetic programming, e.g. Tay & Ho (2008)
- Reinforcement learning, e.g. Zhang & Dietterich (1995)
- Regression trees, e.g. Li & Olafsson (2005)

# Job Shop Scheduling

- Job shop scheduling consists of a set of $n$ jobs that must be scheduled on a set of $m$ machines.
- Each job has an indivisible operation time on machine
- The time in which machine is idle is called slack time,
- Each job must follow a predefined machine order
- Each machine can handle at most one job at a time
- Optimal schedule is the one where the time to complete all jobs is minimal (minimum makespan).

# Example of Job Shop Scheduling

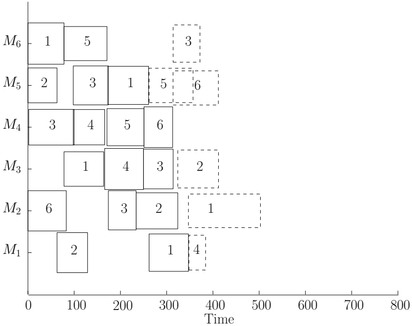


Figure: A schedule being built, the dashed boxes represent six different possible jobs that could be scheduled next using a dispatch rule.

# Dispatching rules for solving JSSP

- Dispatching rules are of a construction heuristics, where one starts with an empty schedule and adds on one job at a time.
- When a machine is free the dispatching rule inspects the waiting jobs and selects the job with the highest priority.
- Most effective single priority based dispatch rules:
  - Most work remaining (MWRM)
  - Least work remaining (LWRM)
  - Shortest processing time (SPT)
  - Largest processing time (LPT)

# Feature selection

| feature | description |
|---|---|
| $\phi(1)$ | processing time for job on machine |
| $\phi(2)$ | work remaining |
| $\phi(3)$ | start-time |
| $\phi(4)$ | end-time |
| $\phi(5)$ | when machine is next free |
| $\phi(6)$ | current makespan |
| $\phi(7)$ | slack time for this particular machine |
| $\phi(8)$ | slack time for all machines |
| $\phi(9)$ | slack time weighted w.r.t. number of operations already assigned |

Table: Features for JSSP

# Generating training data

- Determine the order (sequence) of jobs assigned, at the first available time slot (to the left)
- When job is assigned, new state occurs and features are updated
- At each time step, a good/bad ordinal data pair is only created if final makespan is different.
  - At least one or more optimal solution for each JSSP
  - Sequence representation is not uniquely determined.

# Preference learning

- The preference learning problem is specified by a set of point/rank pairs:
  - Optimal decision: $\vec{z_o} = \vec{\phi}^{(o)} - \vec{\phi}^{(n)}$, ranked $+1$
  - Non-optimal decision: $\vec{z_n} = \vec{\phi}^{(n)} - \vec{\phi}^{(o)}$, ranked $-1$
  - In this study the training set is created from known optimal sequences of dispatch.

# Logistic regression

- Mapping of points to ranks: $\{h(\cdot) : \Phi \mapsto Y\}$
  - $\vec{\phi}_o \succ \vec{\phi}_s \quad \Leftrightarrow \quad h(\vec{\phi}_o) > h(\vec{\phi}_s)$
- Logistical regression: obtain function $h^*$ that can for a given pair $(\vec{\phi}_i, y_i)$ and $(\vec{\phi}_j, y_j)$ distinguish between two different outcomes: $y_i > y_j$ and $y_j > y_i$.
- Problem of predicting the relative ordering of all possible pairs of examples

The surrogate considered may be defined by a linear function in the feature space:

$$h(\vec{\phi}) = \sum_{i=1}^{m} w_i \vec{\phi} = \langle \vec{w} \cdot \vec{\phi} \rangle.$$

# Training size

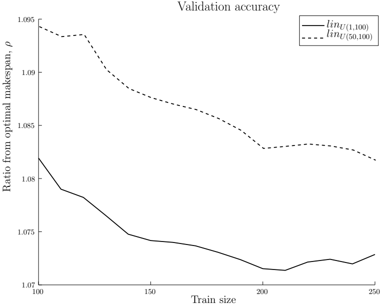


Figure: Deviation from optimal makespan as a function of size of training set. Solid line represents model $lin_{U(1,100)}$ and dashed line represents model $lin_{U(50,100)}$.

# Training accuracy

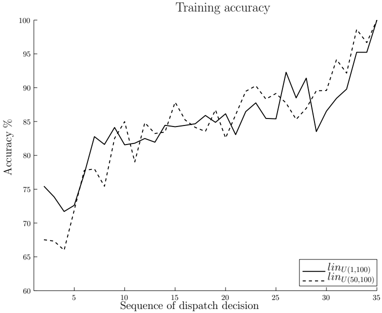


Figure: Training accuracy as a function of time. Solid line represents model $lin_{U(1,100)}$ and dashed line represents data distributions $lin_{U(50,100)}$

# Comparing different dispatching rules

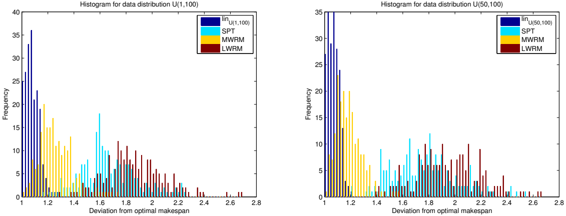


Figure: Histogram of deviation from optimal makespan for the dispatching rules ($lin_{U(R,100)}$), ($SPT$), ($MWRM$) and ($LWRM$). The figure on the left depicts model $lin_{U(1,100)}$, and the figure on the right is of model $lin_{U(50,100)}$.

# Comparing different dispatching rules using ratio from optimality

| $U(1,100)$ | mean | std | med | min | max |
|---|---|---|---|---|---|
| $lin_{U(1,100)}$ | 1.0842 | 0.0536 | 1.0785 | 1.0000 | 1.2722 |
| $SPT$ | 1.6707 | 0.2160 | 1.6365 | 1.1654 | 2.2500 |
| $MWRM$ | 1.2595 | 0.1307 | 1.2350 | 1.0000 | 1.7288 |
| $LWRM$ | 1.8589 | 0.2292 | 1.8368 | 1.2907 | 2.6906 |
| $U(50,100)$ | mean | std | med | min | max |
| $lin_{U(50,100)}$ | 1.0724 | 0.0446 | 1.0713 | 1.0000 | 1.2159 |
| $SPT$ | 1.7689 | 0.2514 | 1.7526 | 1.2047 | 2.5367 |
| $MWRM$ | 1.1835 | 0.0994 | 1.1699 | 1.0217 | 1.5561 |
| $LWRM$ | 1.9422 | 0.2465 | 1.9210 | 1.3916 | 2.6642 |

Table: Mean value, standard deviation, median value, minimum and maximum values using the test sets corresponding to data distributions $U(1,100)$ (above) and $U(50,100)$ (below) .

# Robustness towards data distribution using ratio from optimality

| | model | test set | mean | std | med | min | max |
|---|---|---|---|---|---|---|---|
| #1 | $lin_{U(1,100)}$ | $U(1,100)$ | 1.0844 | 0.0535 | 1.0786 | 1.0000 | 1.2722 |
| #2 | $lin_{U(50,100)}$ | $U(1,100)$ | 1.0709 | 0.0497 | 1.0626 | 1.0000 | 1.2503 |
| #3 | $lin_{U(1,100)}$ | $U(50,100)$ | 1.1429 | 0.1115 | 1.1158 | 1.0000 | 1.5963 |
| #4 | $lin_{U(50,100)}$ | $U(50,100)$ | 1.0724 | 0.0446 | 1.0713 | 1.0000 | 1.2159 |

Table: Mean value, standard deviation, median value, minimum and maximum values for the test sets corresponding to data distributions $U(1,100)$ and $U(50,100)$, on both models $lin_{U(1,100)}$ and $lin_{U(50,100)}$.

# Feature selection

| weight | $lin_{U(1,100)}$ | $lin_{U(50,100)}$ | description |
|---|---|---|---|
| $\bar{w}(1)$ | -0.6712 | -0.2220 | processing time for job on machine |
| $\bar{w}(2)$ | -0.9785 | -0.9195 | work remaining |
| $\bar{w}(3)$ | -1.0549 | -0.9059 | start-time |
| $\bar{w}(4)$ | -0.7128 | -0.6274 | end-time |
| $\bar{w}(5)$ | -0.3268 | 0.0103 | when machine is next free |
| $\bar{w}(6)$ | 1.8678 | 1.3710 | current makespan |
| $\bar{w}(7)$ | -1.5607 | -1.6290 | slack time for this particular machine |
| $\bar{w}(8)$ | -0.7511 | -0.7607 | slack time for all machines |
| $\bar{w}(9)$ | -0.2664 | -0.3639 | slack time weighted w.r.t. number of operations already assigned |

Table: Mean value, standard deviation, median value, minimum and maximum values for the test sets corresponding to data distributions $U(1,100)$ and $U(50,100)$, on both models $lin_{U(1,100)}$ and $lin_{U(50,100)}$.

# Fixed weights vs. varied weights

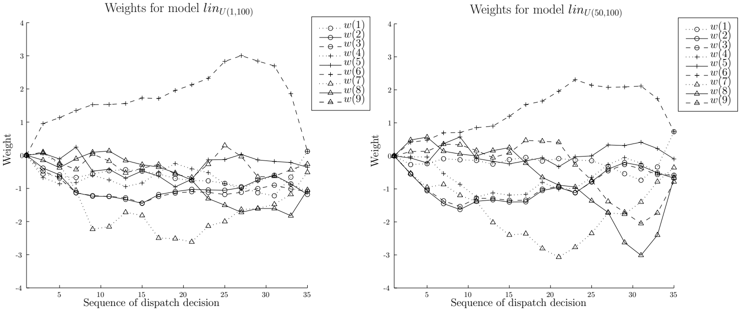


Figure: Weights of features as a function of time, for data distribution $U(1,100)$ (left) and $U(50,100)$ (right).

# Robustness towards data distribution using fixed weights using ratio from optimality

| | model | test set | mean | std | med | min | max |
|---|---|---|---|---|---|---|---|
| #1 | $\bar{lin}_{U(1,100)}$ | $U(1,100)$ | 1.0862 | 0.0580 | 1.0785 | 1.0000 | 1.2722 |
| #2 | $\bar{lin}_{U(50,100)}$ | $U(1,100)$ | 1.0706 | 0.0493 | 1.0597 | 1.0000 | 1.2204 |
| #3 | $\bar{lin}_{U(1,100)}$ | $U(50,100)$ | 1.1356 | 0.0791 | 1.1296 | 1.0000 | 1.5284 |
| #4 | $\bar{lin}_{U(50,100)}$ | $U(50,100)$ | 1.0695 | 0.0459 | 1.0658 | 1.0000 | 1.2201 |

Table: Mean value, standard deviation, median value, minimum and maximum values for the test sets corresponding to data distributions $U(1,100)$ and $U(50,100)$, on both fixed weight models $\bar{lin}_{U(1,100)}$ and $\bar{lin}_{U(50,100)}$.

# Future work

- Overcome problems due to non unique sequence representation of JSSP
- Other learning methods,
  - supervised learning, e.g. decision trees;
  - unsupervised learning, e.g. reinforcement learning;
- Other data distributions and dimensions of JSSP
- Adding due dates to JSSP