# An Integer Programming Model and Heuristic Algorithm for Automatic Scheduling in Synchrotron Facilities

Zahid Anwar, Zhiguo Wang, Chun Wang, Dan Ni, Yaofeng Xu, Yuhong Yan
ENCS Concordia University
Montreal, Canada
{azahid, zhig_wan, cwang, d_ni, yaofeng.xu, yuhong}@encs.concordia.ca

***Abstract*—This paper studies the automatic scheduling problem at the Canadian national synchrotron facility, Canadian Light Source (CLS). An automatic scheduling tool needs to be developed to replace the current manual approach for scheduling experiments on a set of beamlines - resources that generate high-intensity X-rays for use in many kinds of scientific experiments. We present an, Integer programming model for this scheduling activity by formulating it as a problem of *unrelated and paralleled machines with partially overlapping capabilities*. Furthermore a heuristic based approach is used that can save computation time by pruning the search space. Using realistic data sets generated using parameters made available by CLS, we compare the performance of the base line approach that uses ILOG CPLEX implementation of the Integer programming algorithm with one that uses heuristics. The results show that the heuristic approach runs faster than the base-line, but at the cost of producing a less optimal scheduling solution. An obvious advantage of the study presented in this paper is that the automatic scheduling can handle more scheduling conditions and constraints than humans are able to handle manually and can reach optimal solutions. As far as we know, this is the first attempt to propose an automatic scheduling approach for synchrotron facilities like CLS around the world.**

***Index Terms*—Scheduling, Integer Programming, Synchrotron**

## I. Introduction

The CLS facility, operated by Canadian Light Source Inc., is a national science research laboratory for the production of high intensity synchrotron light from the infrared, visible, and ultraviolet to x-ray region of the electromagnetic spectrum and is accessible to scientists and researchers from the academic, government, and private sectors. Today the high-intensity X-rays available at modern synchrotron sources have become an indispensable tool for protein crystallography. Virtually all crystallographic experiments on proteins and other biological macromolecules are conducted at synchrotron facilities. The process works by passing high voltage electricity through a heated cathode producing pulses of electrons. These electrons are accelerated using a linear accelerator to almost the speed of light and periodically injected into a storage ring where they are forced to rotate for hours by means of large bending electromagnets. At each bend the electrons emit synchrotron light which is reflected using mirrors into special vacuum end-stations known as beamlines that are customized to particular frequencies for use in crystallographic experiments. A representative floor plan of the CLS facility is illustrated in figure 1. The beamlines are a shared resource for scientists planning to conduct experiments.

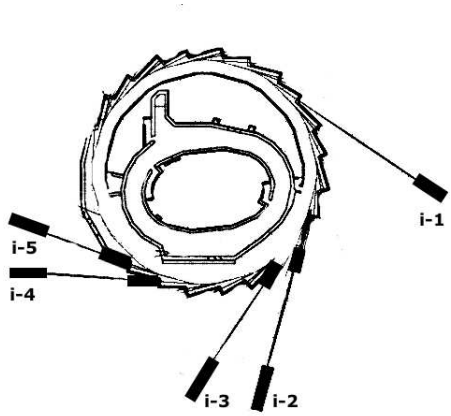


Fig. 1. A Synchrotron facility is characterized by a central electron storage ring and beamlines in the peripheral. There may be multiple beamlines for each frequency group. Three in this case $\{i_1\}, \{i_2, i_3\}, \{i_4, i_5\}$

Currently, the CLS has about 3,000 researchers in Canada and other parts of the world as its user community. The CLS has two calls for proposals each year resulting in a scheduling cycle of 6 months. The proposals that are approved by the peer-review procedure need to be scheduled in the next scheduling cycle. It is assumed that the researchers who submit the proposals know the properties of beamlines and can choose a proper beamline for their experiments. Some experiments can be done by multiple beamlines but some can be conducted only at a certain beamline, or at a certain time period when the beamline is working on a proper working mode. The schedule for different working modes is not considered in our current model. Normally experiments do not need two or more beamlines simultaneously. This problem can be incorporated in our scheduling model, as a case of having paralleled machines (beamlines), but not related, and their capabilities are overlapping for jobs (experiments). In the current manual procedure,

Science Studio project is funded by Canarie Network Enabled Platform Program contract number NEP-01.

the beamlines scientists are responsible for scheduling all the approved experiments. They normally start by scheduling the experiments with higher priority, and try to schedule as many experiments as possible within the beamline operating cycle. As there are many constraints on resource capability, availability, user preferences, as well as priorities to consider, no one has ever been able to check if the manual scheduling results are optimal or not. Each year, the CLS turns away many proposals due to all kinds of reasons, while the user community complains that there are idle gaps on some beamlines.

There are about 30 plus CLS like facilitates [2] around the world. The proposal approval and the scheduling approach are similar to CLS. In order to better utilize such expensive facilities, automated scheduling tools need to be developed. The contribution of our work is two-fold a) the formalization of automated beamline scheduling as a problem of unrelated and paralleled machines with partially overlapping capabilities; b) a heuristic based approach that can save computation time by pruning the search space. we compare the performance of the base line approach that uses ILOG CPLEX implementation of the Integer programming algorithm with one that uses heuristics. The results show that the heuristic approach runs faster than the baseline but at the cost of a less optimal solution.

The remainder of this paper is organized as follows: Section II provides the background on how scheduling automation fits into the larger ScienceStudio project and III discusses related work on scheduling techniques for job scheduling in experimental facilities. Section IV presents a formal description of the problem and outlines the model using Integer programming. Section V shows our implementation via the commercial ILOG CPLEX tool and our heuristic algorithms for optimizing the search time. Section VI describes the details of our model evaluation on realistic test cases. We conclude the paper with a short discussion and future work in Section VII.

## II. Background: The Science Studio Project

The automation of the synchrotron scheduling activities at CLS is part of a larger enterprise application project known as Science Studio. At present synchrotron data collection requires that researchers travel to the synchrotron facility which is cumbersome, results in time delay between the conception and the execution of the experiment and leads to inefficient use of valuable time. Science Studio develops a complete experiment management system [1] allowing researchers to control and observe, from their home base, aspects of research that must be carried out at CLS and other specialised laboratories throughout Canada. Science Studio manages the entire life cycle of the experiment, including sample selection, scheduling, training, control and observation of the experiment and collaborative review of the data. In addition it encompasses an entire workflow management system that allows all involved parties like the scientists, CLS administration as well as shipping, health and safety personnel to track and communicate information about each task involved in the life cycle in a timely manner.

## III. Related Work

Most of the surveys in scheduling theory [4] present parallel scheduling problems holding great theoretical and practical interest. Since the Mac Nuughtons publication [5], the interest of parallel machines scheduling problems has grown increasingly. This is essentially due to the fact of their appearance in a great number of contexts, especially in the computer area for the time management on parallel processors [6], [7]. Theoretical studies in the parallel machines area focus essentially on the search for approximate algorithms with a guarantee of performance. Many heuristics with various criteria have been envisaged in the literature to solve this class of problems [8], [9].

After doing a survey of the synchrotron community [2], we conclude that there is no automated scheduling system in any synchrotron facility. All the facilities have similar proposal approval procedures, regardless how long a cycle is. Generally scientists who are in charge of scheduling experiments on beamlines use e-mail and spreadsheet applications as their primary tools to communicate with the users and manually scratch the schedules on calanders. In order to make their lives easier, the beamline scientists tend to add some constraints to limit the possible combinations they should consider. For example, many facilities reserve a certain beamline only for commercial users, or only for academic users. The Advanced Light Source (ALS)[3] - synchrotron facility in Berkeley California, uses this kind of strategy. Other facilities (like CLS) reserve time slots instead of exclusive beamlines. Furthermore scheduling under conflicting contraints can easily become intractable as the number of users increase.

When we expanded the scope of our investigation to online experiment laboratories in general, we found primarily two types of simplistic *automated* scheduling procedures:

- The system initially collects and sorts all requests, arranging them by considering preferred and unpreferred times.
- The system books available time slots on-the-fly for requesters according to a first-come first-serve policy.

CHEMGAROO [10] an online chemical laboratory catering to 13 universities in Germany uses the second stated scheduling approach. To the best of our knowledge, automated scheduling systems have not been applied to synchrotron facilities scheduling.

## IV. Integer Programming Model

The CLS facility provides a set of $m$ beamlines $I = \{i_1, i_2, ...i_m\}$ for use in $n$ experiments $J = \{j_1, j_2, ...j_n\}$. Experiments are characterized by the following *parameters* that serve as inputs to the scheduling algorithm:

- a **cycle start time** $S_t$ when the scheduling cycle starts.
- a **cycle end time** $T_e$ when the scheduling cycle ends.
- a **release time** $r_j$ where $j \in J$ the earliest possible start time. Before that time, experiments cannot be scheduled due to various reasons e.g. user preferences or preparation of samples (chemical treatment, cutting etc..)

- a **deadline** $d_j$ where $j \in J$ the latest possible finish time. An experiment should be performed between its release time and deadline.
- a **processing time** $p_j$ where $j \in J$ is the time taken to complete the experiment
- a **weight** $w_j$ where $j \in J$ that represents the priority given to the experiment. Many factors can determine the priority for an experiment. For example, the experiments with biological samples have high priority. Similarly commercial experiments have higher priority than academic experiments.
- **eligibility** $e_{ij} = \{0, 1\}$ where $i \in I, j \in J$ is the binary value which depicts the set of eligible beamlines for an experiment.

The problem described above was modelled using Integer programming with the following *decision variables* that are determined by the algorithm subject to a set of *constraints*.

- **start time** $s_j$ where $j \in J$ is the scheduled time for starting the experiment
- **assignment** $x_{ij}$ where $i \in I$ and $j \in J$ is a binary value which depicts which experiment is assigned to which beamline

The following rules represent the constraints that the scheduling has to adhere to:

One beamline *per* experiment:

$$\forall j \sum_{i=0}^{m} x_{ij} = 1 \; s.t. \; x_{ij} \in \{0, 1\}; \tag{1}$$

Start time *after* release time:

$$\forall j \; s_j \geq r_j; \tag{2}$$

Each start time *after* cycle start time:

$$\forall j \; s_j \geq S_t; \tag{3}$$

Start plus processing time less than cycle end time:

$$\forall j \; s_j + p_j \leq T_e; \tag{4}$$

Only eligible beamlines can be selected:

$$\forall i, j \; x_{i,j} \geq e_{i,j}; \tag{5}$$

No overlap of experiment per beamline:

$$\begin{gathered} \forall i, j, k \; s.t. \;\; j \neq k \\ s[i, j] >= s[i, k] + p[k] \vee \\ s[i, k] >= s[i, j] + p[j] \end{gathered} \tag{6}$$

The objective of the scheduling algorithm is to minimize the **total weighted lateness** meaning the sum of time differences between the expected deadline of an experiment and its actual completion time, representing how much the schedule satisfies users' expectations in terms of users' preferred finish times.

Total Weighted Lateness:

$$\sum_{j=1}^{n} w_j.(s_j + p_j - d_j) \tag{7}$$

## V. Heuristic Algorithm

Since job scheduling on unrelated and parallel machines with partially overlapping capabilities is NP hard, algorithms for finding an optimal solution in polynomial time are therefore unlikely to exist. Therefore, a feasible solution is to design a heuristic based algorithm that is inspired by the rules of thumb from real world practices. The principle of a heuristic algorithm is that by using a heuristic function to guide search, we can reach a solution fast with the cost of possibly missing the global optimized solution.

For instance two 'good' experiment-to-beamline assignment heuristics to speed up our search algorithm could be:

1) Schedule the *urgent* experiments first. This can be achieved by sorting them according to *Release time* + *Processing time* - *Deadline* times the *priority*
2) Schedule an experiment on the *least busy* beamline determined by $e[i][j]$. For instance there may be a large number of specialized frequency experiments to be scheduled that have only one eligible beamline to run on. Then experiments with less stringent requirements should be scheduled elsewhere.

In our problem domain, the optimization goal is to minimize the total weighted lateness. This criterion considers both the priority of an experiment (by weight $w_j$) and reducing lateness of served experiments. In the real world, the beamline scientist tries to schedule high priority experiments first. In our algorithm, we schedule the experiments having high weighted lateness first. To avoid conflicts, we dispatch experiments to different beamlines. Currently we do not assign the preference, but the feasibility, of beamlines for an experiment. Therefore, the beamlines are chosen merely by chance.

Algorithm listing 1 shows the heuristic scheduling algorithm. In this algorithm, $Ws$ is the array that holds the weighted lateness for all the experiments. $tmp$ is the array that records the ranks of the experiments on the weighted lateness. For example $tmp[1]$ is the index of the experiment which has the highest weighted lateness. Inside the loop "for i=k to m do", we try to assign experiments, in the order of their ranks, to the first feasible beamline. $brk$ is an array to record the time that is occupied on beamlines, so that we know the time point available for the following experiments. $obj$ is the objective of *total weighted lateness* and is used for comparison with the ILOG CPLEX implementation.

## VI. Evaluation

This section describes an algorithm for generation of realistic data sets pertinent to the Synchrotron scheduling requirements and a comparison of the results produced by both

/*initialize m=size(I) n=size(J) k=1 obj=0 brk[0,...,I]=0*/
**procedure** $HeuristicScheduling(I, J)$
  **for** EACH $j \in J$ **do**
    $Ws[j] \leftarrow w[j].(r[j] + p[j] - d[j])$
  **end for**
  $tmp[j] \leftarrow sortdecending(J, criterion = Ws[j])$
  **for** EACH $j \in J$ **do**
    **for** $i = k\ TO\ m$ **do**
      **if** $e[k][tmp[j]] == 1$ and
      $max(T_s, brk[i], R[tmp[j]]) + P[tmp[j]] < T_e$ **then**
        /* arrange the experiment */
        $x[i][j] \leftarrow 1$
        $s[j] \leftarrow max(T_s, R[tmp[j]], brk[i])$
        $brk[i] \leftarrow s[j] + P[tmp[j]]$
        $obj \leftarrow (brk[i] - D[tmp[j]]) * W[tmp[j]] + obj$
        $k \leftarrow k + 1$
        $break$
      **end if**
      **if** $k > m$ **then**
        $k \leftarrow 1$
      **end if**
    **end for**
  **end for**

**Algorithm 1:** Heuristic Scheduling

Integer programming as well as the heuristic search approach using Gantt charts.

### A. Generation of Test Data

Parameters from the manual scheduling process at CLS were used for generation of test data sets. The proposal evaluation and scheduling process starts when an end user submits a proposal for an experiment. Then, the Canadian Light Source committee evaluates the proposal and puts the experiment into a category which decides its priority. This priority is denoted by the weight $w_j$, the larger the value, the higher the priority. All proposals must be received before a deadline to be scheduled into the next six-month cycle.

Not all experiments can be conducted on all beamlines. Some experiments can only be executed at a certain frequency that is particular of certain beamlines. The required frequency must be provided by the end user at the time of the proposal submission. The parameter $e_{ij}$ controls which experiment $j$ can be run on a certain beamline $i$ by taking the required frequency into consideration.

The goal is to minimize the *Total Weighted Lateness* while meeting the constraints described earlier in Section IV. The output of the heuristic algorithm should give a value of the *Total Weighted Lateness* as well as the assignment of beamlines to experiments with start times, processing times and end times.

We introduce an algorithm shown in listing 2 to generate $G$ testing data sets.

- rand(Num1, Num2) generates a random integer between [Num1, Num2] using uniform distribution
- rand[Num1, Num2] generates a number that is either Num1 or Num 2

**procedure** $GenerateEvaluationData(G)$
  **for** EACH $k \in G$ **do**
    $S_t \leftarrow rand(0, 10)$;
    $T_e \leftarrow S_t + rand(500, 600)$;
    $M[k] \leftarrow rand(1, 4)$; /*number of beamlines*/
    $N[k] \leftarrow rand(1, 100)$; /*number of experiments*/
    **for** EACH $j \in [0, \ldots, N[k]]$ **do**
      $r_j[k] \leftarrow rand(1, 500)$;
      $d_j[k] \leftarrow r_j[k] + p_j[k] + rand(0, 20)$;
      $w_j[k] \leftarrow rand(1, 100)$;
      $p_j[k] \leftarrow rand(1, 50)$;
      **for** EACH $i \in [0, \ldots, M[k]]$ **do**
        $e_{ij}[k] \leftarrow rand[0, 1]$; /*either 0 or 1*/
      **end for**;
    **end for**;
    $/*(M[k], N[k], \{r_j[k]\}, \{d_j[k]\}, \{w_j[k]\},$
    $\{e_{ij}[k]\}, \{p_j[k]\}, S_t, T_e) is\ a\ data\ set*/$
  **end for**

**Algorithm 2:** Generating evaluation data

For purposes of presentation and easy comprehension, a constraint relaxation we made for the results shown in the next section was $\forall j, \sum_{i=0}^{m} e_{ij} \geq 1$. (to assure: for each beamline, there must exist a experiment that it can accept)

### B. Results

The scheduling algorithm was tested using six different generated data sets of which the results of the first three are compared in detail for the benefit of the reader. ILOG CPLEX OPL Development Studio 6.1.1 was used to implement the Integer programming algorithm and VBA was used for the data generation. Tables I, II and III show the generated data sets for runs 1, 2 and 3 respectively.

| Parameter | Value |
|---|---|
| $M$ | 2 |
| $N$ | 3 |
| $R$ | [1,1,2] |
| $D$ | [8,15,5] |
| $W$ | [4,5,1] |
| $P$ | [10,4,3] |
| $E$ | [[0,1,0],[1,0,1]] |
| $S_t$ | 1 |
| $T_e$ | 100 |

TABLE I
INPUT DATA FOR RUN 1

The results for table I are displayed in figure 2. The schedule returned by ILOG CPLEX and the heuristic algorithm were incidently identical with experiment 2 running exclusively on beamline 1 and experiments 1 and 3 scheduled back-to-back on beamline 2.

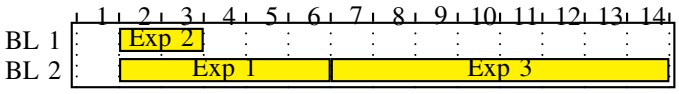


Fig. 2. Run 1: Output of ILOG CPLEX OPL and Heuristic is the same

Results (figure 3) for input shown in table II are more interesting. Because of the flexibility in deadlines the heuristic algorithm was a lot less conservative than ILOG CPLEX when fixing the schedule. This is evident from the fact that the overall run for the heuristic took 162 hours owning to beamline 2 whereas the longest sequence of experiments (beamline 3) using ILOG CPLEX took only 138 hours.

| Parameter | Value |
|---|---|
| $M$ | 3 |
| $N$ | 9 |
| $R$ | [38,6,53,60,63,28,59,23,25] |
| $D$ | [98,69,88,73,91,59,121,83,86] |
| $W$ | [31,78,2,77,82,71,5,42,87] |
| $P$ | [40,44,19,3,15,14,42,46,50] |
| $E$ | [[1,0,0,1,1,1,0,1,0],[1,0,1,1,0,0,0,0,1],[0,1,0,1,0,0,1,1,1]] |
| $S_t$ | 6 |
| $T_e$ | 535 |

TABLE II
INPUT DATA FOR RUN 2

Finally figure 4 shows significantly poorer performance in finding a suitable schedule at the cost of search speed. Both beamline 1 and 2 experiment runs took longer than their ILOG CPLEX counterparts.

| Parameter | Value |
|---|---|
| $M$ | 2 |
| $N$ | 12 |
| $R$ | [59,99,11,100,8,70,12,45,3,56,87,71] |
| $D$ | [97,125,18,167,42,95,45,54,25,100,123,108] |
| $W$ | [4,1,1,3,1,4,2,4,4,1,2,4]; |
| $P$ | [34,22,5,48,18,11,23,1,16,24,24,23] |
| $E$ | [[1,0,0,1,1,1,0,1,1,1,1,1],[0,1,1,0,1,1,1,1,1,1,1,1]] |
| $S_t$ | 8 |
| $T_e$ | 520 |

TABLE III
INPUT DATA FOR RUN 3

### C. Discussion

The results show that the schedule produced by the heuristic is a small order of magnitude larger in terms of the *Max End Time* and the *objective* function of total weighted lateness and where *Max End Time* is the completion time of the very last task in the entire run. A comparison of the two indicators is summarized in table IV

The advantage the heuristic gives us over ILOG CPLEX is in running time. What was consistently observed was that if there are more than 5 beamlines or more than 20 experiments then it takes ILOG CPLEX more than 2 days to process and produce the schedule. Having a lot of eligible beamlines characterized by a large number of 1s in E[i][j] also increases the running time. On the other hand the heuristic program always finishes in a few seconds.

| | **ILOG CPLEX** | | **Heuristic** | |
|---|---|---|---|---|
| | Max End Time | Objective | Max End Time | Objective |
| Case 1 | 14 | -29 | 14 | -29 |
| Case 2 | 138 | -4232 | 162 | 14831 |
| Case 3 | 165 | -105 | 204 | 1854 |
| Case 4 | 127 | -103 | 177 | 153 |
| Case 5 | 91 | -189 | 124 | 107 |

TABLE IV
COMPARISON OF SCHEDULES PRODUCED

## VII. CONCLUSION

In this paper we model the resource scheduling activity in scientific experimental facilities like CLS as an Integer programming problem and develop algorithms for its automation. Heuristics are presented to speed up the search for an optimal solution. The algorithms are tested on realistic generated inputs mimicking the actual proposal submission process used for scheduling beamline experiments at CLS.

The commercial ILOG CPLEX tool is used to implement the Integer programming scheduling algorithm. While it is observed that the schedule produced provides the best utilization and minimizes the total weighted lateness, the running time increases exponentially for large inputs. This is resolved by the use our heuristic algorithm that typically takes a running time of a few seconds to produce a schedule at the cost of a slightly higher objective of total weighted lateness.

In the future we would like to design a scheduling language which would allow scientists to convey their requirements to the algorithm automatically. So far we have assumed that resources are always available and the requirements of scientists do not change once the scheduling process is complete. This is not completely realistic e.g. beamlines may go down for maintenance or experiments might get delayed. We would like to extend our model to include these dynamic scheduling requirements where the inputs may change after the initial scheduling.

## ACKNOWLEDGMENT

The authors would like to thank Elder Matias and Lavina Carter from CLS, Stewart McIntyre from UWO and John Haley and Chris Armstrong from IBM, for the detailed discussions and help in clearing up CLS scheduling requirements.

## REFERENCES

[1] Canadian Light Source, *Remote Instrument Control*, http://www.lightsource.ca/ operations/ remote_access_project/
[2] Wikipedia Article, *Synchrotron*, Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Synchrotron. March, 2009.
[3] ALS User Services Office. *The Advanced Light Source*, Lightsources.org. http://www.als.lbl.gov/. Feb, 2009.

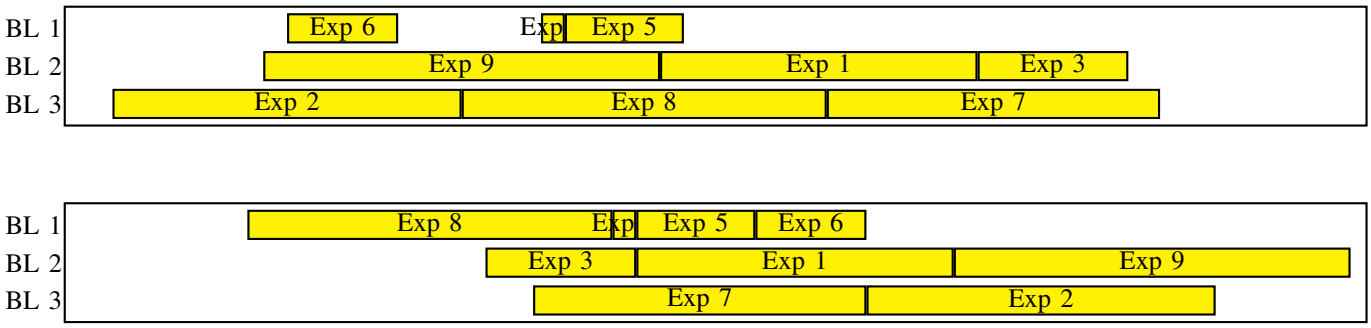


Fig. 3. Run 2: Output of ILOG CPLEX OPL (Top) and Heuristic (Bottom)

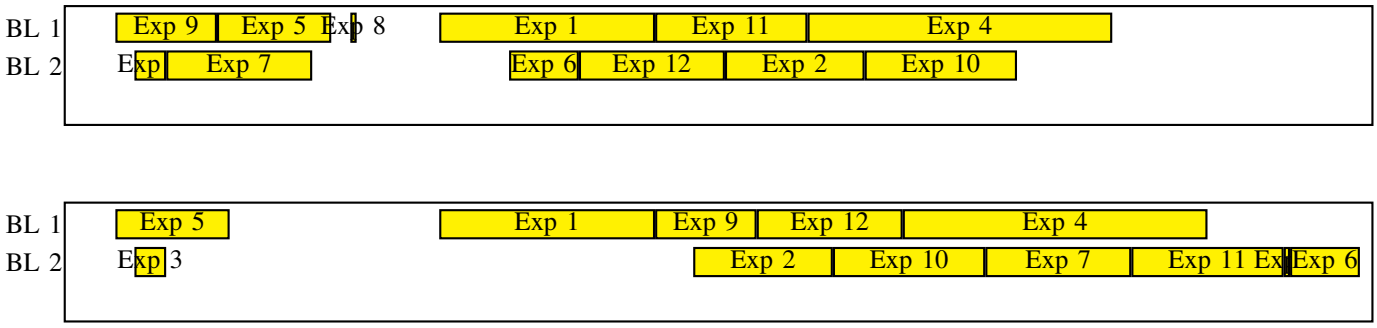


Fig. 4. Run 3: Output of ILOG CPLEX OPL (Top) and Heuristic (Bottom)


[4] E.L. Lawler and J.K. Lenstra and A.H.G. Rinnooy Kan. and D.B. Shmoys. *Sequencing and scheduling: Algorithms and complexity.* Technical Report BS-R8909, Centre for Mathematics and Computer Science Amsterdam, 1989.

[5] R.M. MacNaughton. *Scheduling with deadlines and loss functions. Management Science.* 6(1):1-12, 1959.

[6] J. Blazewicz. *Selected topics in scheduling theory. Ann. Discrete Math.*, 31:l-60,1987.

[7] T.C.E. Cheng and C.C.S. Sin. *A state-of the- art review of parallel-machine scheduling.* European Journal of Operational Research, 47:271-292, 1990.

[8] R.T. Sumichrast and J.R. Baker. *Scheduling parallel processors: An integer linear programming based heuristic for mininimizing setup time.* International Journal of Production Research, 25(5):761-771, 1987.

[9] K.C. So. *Some heuristics for scheduling jobs on parallel machines with setups.* Management Science, 36(4):467-475, 1990.

[10] FIZ CHEMIE, *CHEMGAROO - Educational Systems*, http://www.fiz-chemie.de/ en/home/products-services/ education-training/chemgaroo.html, 2009.