Teaching Constraint Programming in the SFI CRT-AI Program

Janet Choi ✉
The SFI Centre for Research Training in Artificial Intelligence, School of Computer Science and Information Technology, University College Cork, Cork, Ireland

Barry O'Sullivan ✉
Insight Centre for Data Analytics, School of Computer Science and Information Technology, University College Cork, Cork, Ireland

Helmut Simonis ✉
Insight Centre for Data Analytics, School of Computer Science and Information Technology, University College Cork, Cork, Ireland

Abstract

In this paper we describe our experience with teaching Constraint Programming and Optimization to PhD students as part of the Irish national CRT-AI doctoral training program. The course is delivered as a one-week training period, where students from five Irish universities come to Cork. This is one element of a five-week program, with each week delivered at one of the five participating sites. We have given this course four times in 2019, 2021, 2022, and 2023 so far, with modifications of the design and delivery due to the Covid epidemic.

2012 ACM Subject Classification Social and professional topics → Model curricula; Theory of computation → Constraint and logic programming

Keywords and phrases Constraint Programming, Teaching, Optimization

Digital Object Identifier 10.4230/LIPIcs.WTCP.2023.4

Acknowledgements This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 12/RC/2289-P2 at Insight the SFI Research Centre for Data Analytics at UCC, which is co-funded under the European Regional Development Fund. This work was also conducted with the financial support of the Science Foundation Ireland Centre for Research Training in Artificial Intelligence under Grant No. 18/CRT/6223.

1 Introduction

We provide a Constraint Programming and Optimization course as a one week, mandatory training module for students in the first year of their structured PhD programme, who connect in Cork from 4 other universities in Ireland. This is a unique opportunity for PhD students who visit each of the participating sites for discipline specific AI training. This includes a weeks training in Natural Language Processing, Computer Vision, Personalisation and Recommender systems, Machine Learning and Constraint Programming.

When we designed the course content in 2019, we made the following design choices focused on the following elements:

- Only a few students will focus on Constraint Programming as their main topic, we therefore concentrate on Modelling and Decision Support, rather than the implementation side of solvers.
- We use MiniZinc [2] as the tool to be used by the students. While most students are familiar with Python, and other modules in the CRT-AI program are using Python for team assignments, we wanted to use a modelling tool, rather than a programming language as the basis for the course.

- We want to provide some intuition on how Finite Domain Constraint Programming works, but based on examples rather than formal definitions and algorithms.
- Reliance on lectures, and course content alone, without a practical component, can lead to poor learning outcomes. The students should engage in the topic by solving a non-trivial assignment, based on a real-world (like) scenario.
- Students should work in teams to solve the assignment. This not only improves the quality of the solutions obtained, but also provides an opportunity for students to build upon their networks, by interacting with fellow researchers from across the country during in-person training.
- Students should present their results on the assignment at the end of the week, the presentation forming part of their assessment, but also giving some training in creating and presenting results to an audience.
- As students are visiting Cork for the one week training module, they are less constrained by their "normal" work and environment, and can focus on the topic. In that sense, the training course is more like a Summer School or an industrial training program than a academic course taught during a semester.

2 The SFI CRT-AI Program

The SFI Centre for Research Training in Artificial Intelligence (CRT-AI) was established in 2019 as one of the national research training centres to provide training for PhD students in Artificial Intelligence at different universities in Ireland.

The mission of the centre is given on the website (`https://www.crt-ai.ie/`).

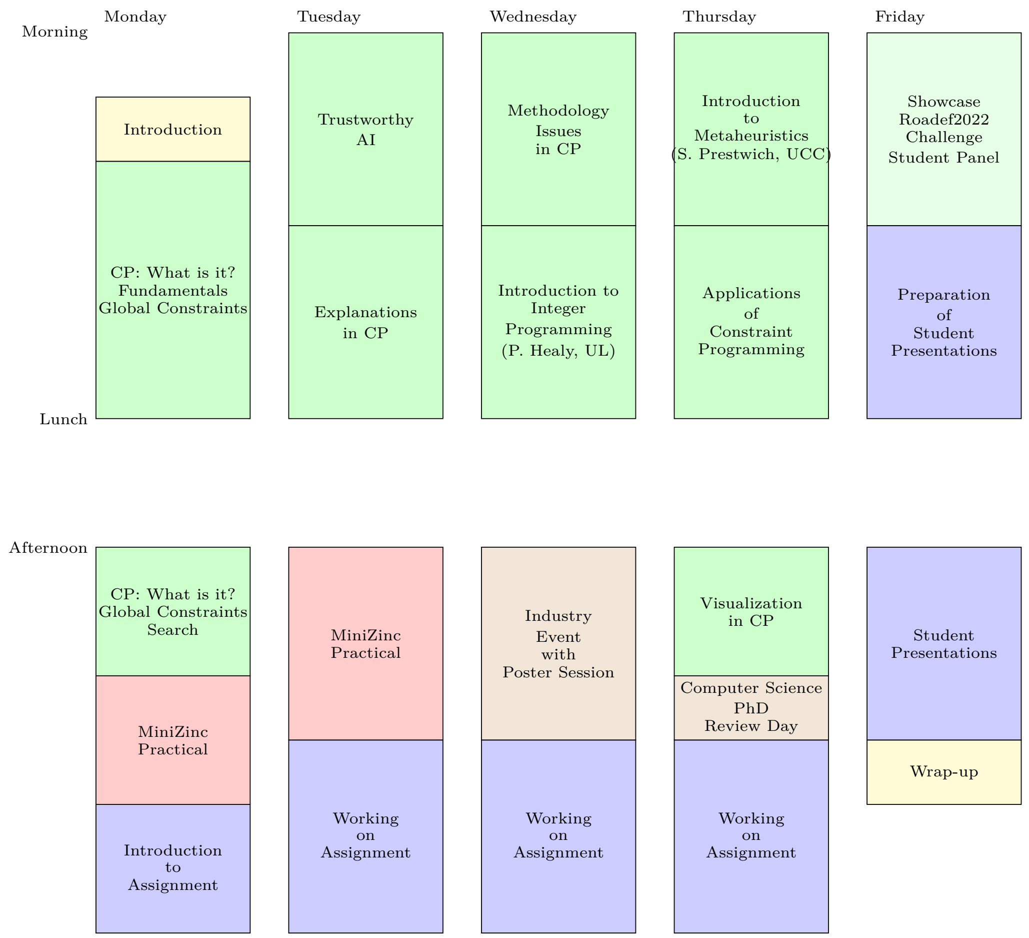
> This centre aims to create an internationally connected and globally recognised centre of excellence for the training of postgraduate students and the up-skilling of industry-based staff in key technical topics in artificial intelligence. The initiative will provide training in areas related to ethics in artificial intelligence and data analytics, as well as challenges in fairness and transparency of advanced data-driven applications. The SFI Centre for Research Training (CRT) brings together supervisors working across the full spectrum of AI techniques from knowledge representation and reasoning, machine learning, data mining, computer vision, natural language processing, optimisation and decision-making, robotics, and autonomy. The CRT will focus strongly on the development of AI solutions in domains such as smart buildings, mobility and transportation, autonomous vehicles, public service delivery, manufacturing, enterprise, cybersecurity, climate change and environment, agriculture, marine, food production, and natural resources.

Our team in the School of Computer Science at University College Cork provides the training in optimization and decision making, with a special focus on Constraint Programming. The students in this program do not all have a Computer Science background, some of them are trained in the arts or law, for example. We can therefore not assume too much background knowledge, in particular many of the students will not have seen Constraint Programming before.

3 CP Week

The programme is delivered during one week from Monday to Friday, with a social event to bring the cohort together on Sunday evening prior to the module start. Depending on the order of the training weeks (which changes from year to year), the cohort of students may have met before at other training weeks, or may be meeting for the first time. Introductions by the students on their background and current work start up the meeting on Monday. The overall structure of the week in 2023 is given in Figure 1, which is similar to previous events in 2019, and 2022. We describe the changes required for a virtual format of the week using in 2021 later in the paper.

Figure 1 Structure of Constraint Training Week in 2023



Lectures are shown in green, practicals in red, and student hands-on training in blue. We also have two additional events, shown in brown, and the introductory and wrap-up sessions colored in yellow.

We now discuss some of the elements of the week in more details, leaving the core CP lectures for the next section.

Practical The practical provides a mixture of presentation and hands-on training, giving most students their first exposure to MiniZinc. The practical covers the basics from installing the tool, elements of the MiniZinc language, running and choosing the back-end solver, and understanding error messages and fixing problems.

Trustworthy AI We include some lectures on current research activities at UCC in the program. Trustworthy AI is a focus of the research at the Insight Centre at UCC, and is also one of the key elements of the CRT-AI program.

Explanations in CP The talk on Explanations in CP is another overview of current research at UCC, which also is of interest as part of the general push towards explainable AI, which is important for all students in the course.

Integer Programming Patrick Healy of the University of Limerick gave a lecture on Integer Programming. The students are already familiar with the MIP back-end solvers in MiniZinc, this talk provides some intuition how these solvers work.

Metaheuristics Steve Prestwich of UCC provided an introduction to meta-heuristics and local search. It would be nice to incorporate this area more closely into the course and practical training, but the use of MiniZinc as the training platform limits choices for this.

Industry Meeting The students of the CRT-AI program must find an industry placement of three to six months as part of their overall training. We organize a meeting of students and potential industry sponsors from the Cork area as part of the training week. This also involves students of earlier cohorts of the training program, who give poster presentations of their work, and can provide some experience with their own industry placement.

CS Review Day The school of Computer Science organized their yearly PhD student review during our training week. This event involves all current PhD students presenting posters to reviewers and the overall department. We made room in the schedule for our students to look at the posters in the review meeting, on one side to get a picture of the type of work done at UCC, but also to experience this type of review as a visitor, before presenting their own work in a future review event.

Showcase An addition to the program this year was a student-led panel on a showcase application. A team of PhD students and senior researchers participated in the Roadef/Euro Challenge 2022 `https://www.roadef.org/challenge/2022/en/index.php` on planning the truck-based transport of parts from suppliers to factories for Renault. Some of the students involved presented their experience with the project, the lessons learned and the impact on their own PhD research.

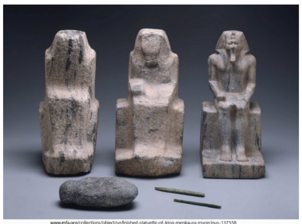
4 Fundamentals of Constraint Programming

The core modules of the training program are a number of lectures on Finite Domain Constraint Programming, based heavily on earlier work for an Elearning Course [3] on the constraint programming language ECLiPSe (`https://eclipseclp.org/ELearning/index.html`). The course tries to introduce the core concepts of domain variables, propagation and search on some well-known examples. It presents CP as a subtractive process (see Figure 2), removing infeasible values leading to a solution.

4.1 Finite Domain Constraint Solving

The first part of the course deals with the concepts of models, finite domain variables, constraints, and propagation, exemplified on the SEND+MORE=MONEY puzzle. It shows the reasoning process in detail (Figure 3), going through all the steps to eliminate infeasible values from the domains. It does not formally describe the propagation reasoning, but shows the reasoning in sufficient details for a human to follow. While we probably never want to go to this level of detail when solving a model, it shows how much propagation is obtained by each of the constraints of the model, and why some search is still required with the standard level of reasoning.

A Subtractive Process



"Oh, bosh, as Mr. Ruskin says. Sculpture, per se, is the simplest thing in the world. All you have to do is to take a big chunk of marble and a hammer and chisel, make up your mind what you are about to create and chip off all the marble you don't want."-Paris Gaulois.

Source: `https://quoteinvestigator.com/2014/06/22/chip-away/`

Figure 2 CP as a Subtractive Process

Propagation of equality (Iteration 1)

$$\underbrace{91 * E^{2..8} + 10 * R^{2..8} + D^{2..8} = 90 * N^{2..8} + Y^{2..8}}_{204..728}$$

$$N \geq 3 = \lceil \frac{204 - 8}{90} \rceil, E \leq 7 = \lfloor \frac{728 - 22}{91} \rfloor$$

Figure 3 Propagation Example

4.2 Global Constraints

The next part of the course introduces global constraints, using the example of the `alldifferent` constraint for the Sudoku puzzle. We present different levels of consistency, starting with forward checking obtained by the binary decomposition, and then introducing bounds and domain consistency by examples. We compare the results obtained on an example Sudoku (Figure 4), but do not detail the algorithms used. While the ECLiPSe training course introduces the methods used (bi-partite matching, connected components, flow models), we think that this material needs too much time to discuss, and is not really required in a course on modelling.

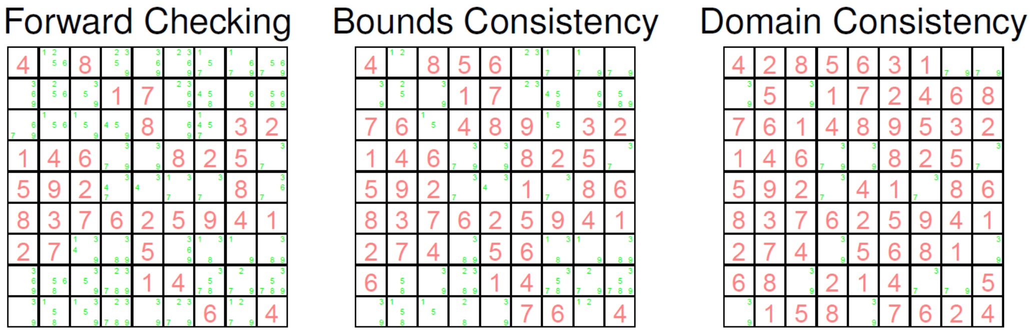


Figure 4 Consistency Level Comparison

4.3 Search

The last part of the course directly derived from the ECLiPSe Elearning course handles the impact of search, and ways to improve the stability of the solver for finding solutions to families of problems. We use the classical N-Queens problem as an example, and show both the resulting search tree (left) and the corresponding propagation (right) using different search annotations (Figure 5).

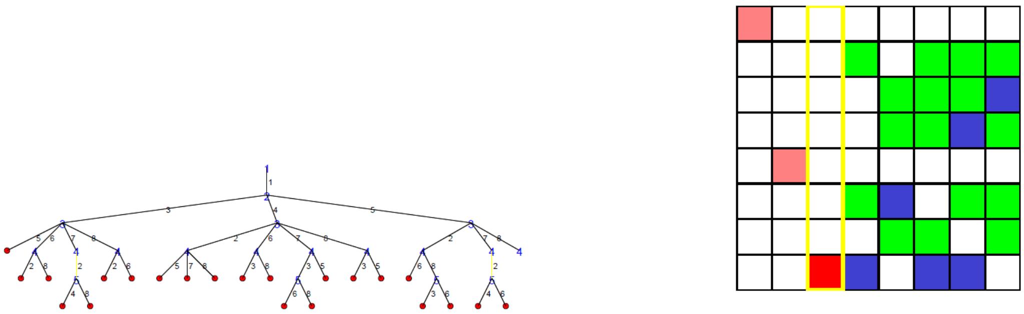
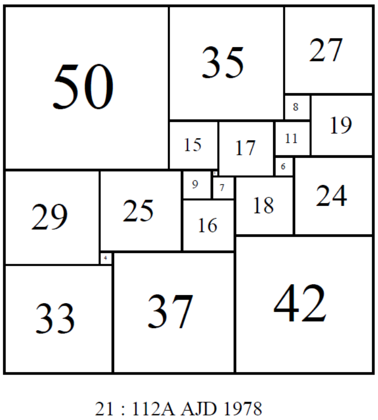


Figure 5 Search and Propagation

While this gives a useful indication what is possible, it also suffers from the limitation of search in ECLiPSe, where some of the most powerful strategies are not available. On the other hand, we do not have an API to show the search behaviour of a more modern solver like Chuffed in MiniZinc at the same level of abstraction.

4.4 Elements of MiniZinc

The next presentation focuses on MiniZinc, and is used in the practical session on Monday afternoon. We discuss the core language, the IDE, and the different back-end solvers, and then show some example programs in MiniZinc, like the Square Packing Problem in Figure 6, which also serves as an introduction to the `cumulative` and `diffn` global constraints. We also show the MT10 job shop scheduling problem, to introduce some concepts of scheduling that will be required in the programming challenge.




```
constraint forall (i in S)
    (x[i]+size[i]<=box);
constraint forall (i in S)
    (y[i]+size[i]<=box);
constraint diffn(x,y,size,size);
constraint cumulative(x,size,size,box);
constraint cumulative(y,size,size,box);

solve satisfy;
```

Figure 6 MiniZinc Square Packing Example

4.5 Applications

So far the course material was focused on small scale examples and puzzles. While these are very useful to understand the concepts of models and propagation, we have to be careful to avoid the impression that CP is only about such small scale problems. The talk on applications gives an overview of different types of applications developed at Insight/UCC, including

- Scheduling
- Health Care Capacity Management
- Service Planning and Scheduling
- Constraint Acquisition

Figure 7 shows a slide from the service planning application (based on work with UTRC), where a clustering step is the first element of a problem decomposition leading to a scheduling system dealing with unforeseen events, while maintaining an overall monthly plan.

While there are of course very interesting CP Applications developed elsewhere, we can present details and issues for our own problems that may not be available in high-level presentations of other applications.

4.6 Methodology

The next talk deals with a methodology for developing constraint or more general optimization applications, considering core elements of requirements capture, data management, and overall use case handling. Figure 8 shows a general scheme that describes the design of CP

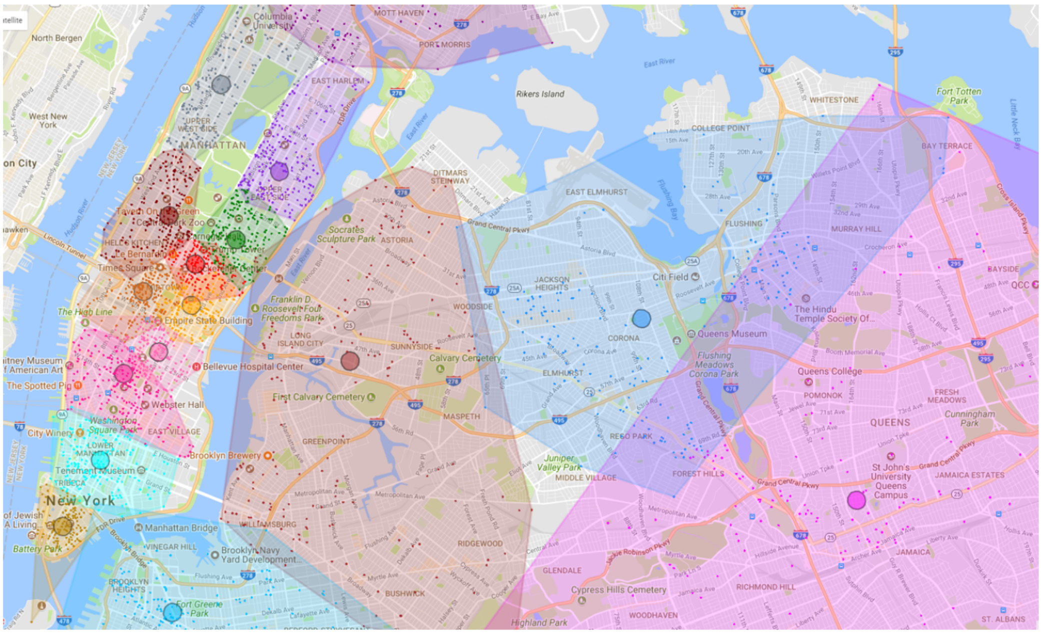


Figure 7 Application Example: Service Areas with Clustering

applications that react to changes in the environment, as well as the partial implementation of solutions obtained in previous runs of the model.

While for many students this section may be too advanced and application focused, we found that nearly all PhD students struggle to understand the importance of these topics, and under-estimate the time and effort required to resolve the resulting issues.

This talk is based on a lecture given at the 2018 ACP Summer School in Jackson/Wyoming (`https://school.a4cp.org/summer2018`), which served as a testing ground for much of the material used in this course.

4.7 Visualization

The final lecture of this course is on visualization, based on a tutorial developed by H. Simonis and G. Tack for CP 2021 [4], with video available at `https://www.youtube.com/playlist?list=PL97NT99ttj2CnvGBFdDTawTnNXAFrf4w-`.

Visualization is one of the main techniques to understand the results of constraint models, and is heavily used in the course material itself. The tutorial gives an overview of the different tools and approaches available (as an example, see Figure 9), and links them back to development methodology questions. While the visualization tools in MiniZinc are improving, it is still too complicated to produce simple visualizations with minimal effort as part of the challenge development. Increasingly students are able to use existing Python libraries to visualize the results of a MiniZinc program, using the JSON output format of MiniZinc for data exchange.

5 Student Project

We decided early-on that the course should include some hands-on training based on a practical problem, given in textual form. This means that the students are not given a

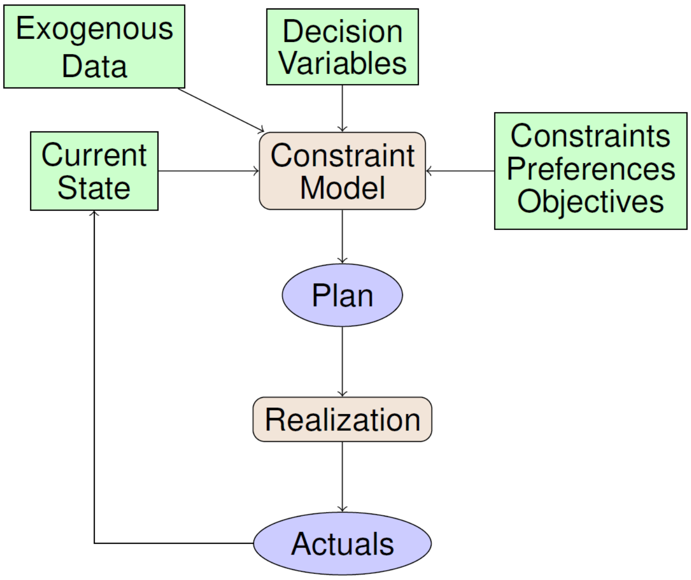


Figure 8 Methodology Example

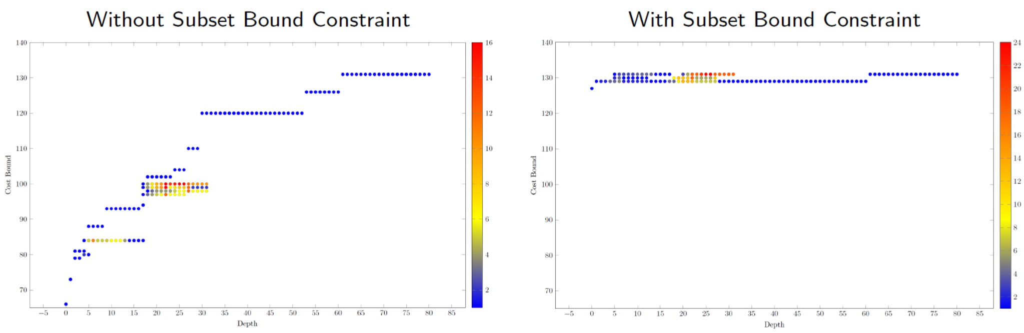


Figure 9 Visualization Example

standard problem formulation (like a puzzle or combinatorial design) to solve, but a somewhat specialized problem based on a real-world example. Appendix A gives the full description for the problem of 2023, for every year we select a new challenge.

2019, Interview Assignment and Scheduling For this problem, students must schedule interviews of a group of students with companies, based on preferences of participants. The initial version is an assignment problem, but later variants change the problem structure to make it more difficult to solve.

2021, Test Scheduling Problem This problem is a scheduling problem based on problem 073 of CSPLib `https://www.csplib.org/Problems/prob073/`[1]. Different types of resource and precedence constraints need to be defined, while searching for an optimal solution.

2022, Assembly Line Balancing The example is based on the Simple Assembly Line Balancing problem in the literature (`https://assembly-line-balancing.de/`), presented as an issue for the Model T assembly at the (now defunct) Ford factory in Cork.

2023, Port Berth Allocation The problem is to plan where to berth ships arriving in a harbor, allowing for restrictions on the type of berth used, the size of ship arriving, and additional constraints about resources and constraints like tides. The problem changes from a pure assignment problem to include scheduling elements as well.

5.1 Problem Description

The problem is given in textual form, trying to avoid giving any hint about the model to be used. The students need to understand the problem itself, and then come up with a formulation of the problem as a model with variables and constraints. Given the different backgrounds of the students, this can be a major challenge for some teams. The teachers are available throughout the study sessions, and can provide hints on how to approach the problem, and how to solve some of the challenges interacting with MiniZinc. While some students are able to solve the problems without further help, the majority of teams need some support at some point in time. Given the limited amount of study time available, it is not always clear when to intervene in a discussion with a helpful hint.

5.2 Changing Problem

It is important to present a new problem every year, as students obviously discuss last year's problem when talking to earlier student cohorts. This creates an additional challenge in setting up the course every year, with the overall effort probably requiring two weeks of preparation each year. As part of the problem development an example solution is created, that is made available after the event.

5.3 Presentations

Student teams not only have to solve the problem, they also must present their solutions as part of an overall assessment at the end of the week. The focus here is not so much on grading success or failure for each task of the challenge, but rather to show the engagement with the material, approaches tried and possibly abandoned, and lessons learned during the process.

6 Different Years

We have run the CP Week over a period four years, with different cohorts of students. The main structure of the week has stayed the same, but some of the events and lectures for more advanced materials have changed over time. The main exception was the course presented in 2021, which was given as a virtual event, with a reverse lecture mode.

6.1 Structure of 2021 Week

The main challenges to presenting the course as a virtual event were
- Not all students were in-country, some were still at home in a different time zone.
- Student interaction was limited by available tools.
- Support of the student project was more difficult.

In answer to the first challenge, we presented the lectures as pre-recorded videos that students could watch at a time of their own choice. The time slots normally reserved for lectures were made available for student questions, and a more interactive online discussion of the material.

Due to the Covid restrictions in place at the time, students were not allowed to meet in teams, even with other students at their home university. This made interaction and cooperation during the project phase much more challenging, relying on tools available to the students.

Compared to the normal pattern of holding all sessions in a lab, it was much more difficult to see which students or student teams were struggling with the assignment.

7 Student Feedback

While there is a feedback mechanism with a questionnaire at the end of the course, the student numbers are perhaps too small to allow a formal statistical analysis. The following feedback was received multiple times:

- The CP week compares favourably to some of the other training weeks at other universities, presenting other topics like Machine Learning, or Natural Language Processing. The reasons given were the interactive elements of the training, and the link to practical example uses throughout the course.
- The use of MiniZinc presents a challenge to many students, especially those with previous programming experience. The challenge of writing a model of the problem, rather than an algorithm, was a difficult concept to grasp.
- MiniZinc itself was criticized for its rather complex syntax, and limited debugging capability.
- Some students complained that the program was "too intense", while others enjoyed the challenge presented.

Some specific feedback was collected from the 2019 Constraint week:

- The lecture was really good. The practical session was also well organised and suitable for all level. The only (very tiny) issue in day 1 is that we found some difficult to find the way to the building. It should have had good guidance showing how to get to the building.
- It was quite intense, which is good, because it motivates really, though I felt there could have been a bit more time assigned for our project.

- The lectures were a bit long so was very tired towards the end.
- Although the topic was not the main area of the most students, the lecturers could present it with examples that were interesting for everyone
- Maybe bring in industry experts from large companies for a practical session on solving real world problems?
- The only thing that I think 2 hours for a lecture is too much.
- The material of the second day was not relevant to the first day!
- The whole day was really interesting, especially the problem given to be solved. It was neat and punctual.
- I think the reception was perfect. The fields of lectures were very useful. It was very good that we had collaboration as members of teams and I think this improves our teamwork skills. The only problem I had was that the courses were very intensive and I did not have enough time for studying courses materials.

A student survey provided some evaluation of the course material by the students, two examples are shown in Figure 10 and 11.

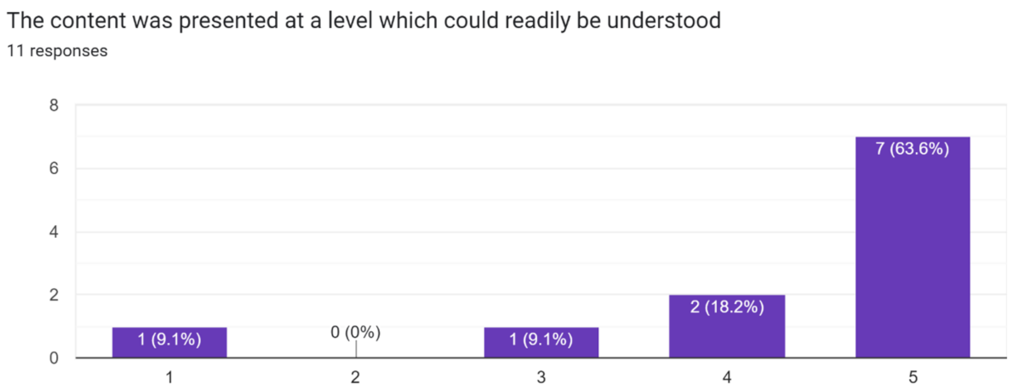


Figure 10 Student Survey: Is the material understandable?

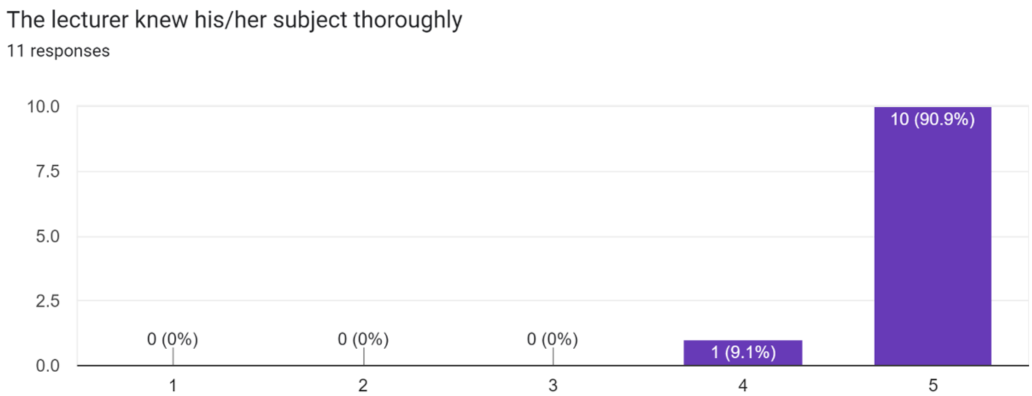


Figure 11 Student Survey: How the lecturers know their material?

The limited number of students and survey replies limit the ability to make a meaningful statistical analysis.

An interesting suggestion by participants from different years during a recent round-table discussion was to involve students from earlier cohorts in the practical and challenge sessions at tutors, as this might lead to a better understanding of questions, and may ease communications.

8 Lessons Learned and Conclusions

We have now run the CP week for four years, with different cohort sizes, and students of different background. What are the lessons we learned?

- First and foremost, the format of the CP week as a week long event seems to work quite well. Students are motivated and not too distracted by other demands on their time, so that a lot of material can be covered in one week.
- The experience of running the course as a virtual event in 2021 highlighted the advantages of an in-person event. While some students adapted very easily to online presentations, more struggled with the format, and had greater difficulty to make progress with their assignment. While pre-recorded lectures may be a bonus overall, the concentrated time frame of the one week format did not give a lot of opportunity to revisit a lecture online.
- While there is not a lot of time dedicated to the fundamentals of Constraint Programming, students do learn the essential elements of CP, and get an impression of the variety of problems that can be addressed by it. Some introductory material may need to be repeated a few times in the lectures and the practical, but the hands-on sessions ensure that students have understood the key concepts.
- Perhaps surprisingly, some students with the most programming background, but no previous experience of modelling languages, struggle the most with the concept of a model and variables, in contrast to user written assignments and classical control structures. Some of the students without computer science background seemed to grasp these fundamental concepts more easily.
- Using a modelling language like MiniZinc helps with making the distinction between models and conventional programs, but it is another syntax to learn, and another IDE to install.
- We have seen the first student use ChatGPT in the course in 2023, not to generate a program, but to find a bug in a manually written program. The resulting interaction showed that ChatGPT did understand MiniZinc to some extent, and correctly highlighted the problem in the user program. We should perhaps test our problem descriptions against generative AI before the course is run, to ensure that the AI does not immediately solve the problem.
- MiniZinc provides a checker mechanism, where students can check their solution against an automated process to highlight missing or mis-understood constraints. This seems to be used to advantage in on-line MOOC to reduce the dependence on manual grading. We have not been providing such checkers for the students. On one side a checker must have a fixed format in which the answer is provided, and we do not force students to use such a fixed format. Coming up with different representations of a problem is a key learning experience. On the other hand, this approach requires more work on behalf of the teachers, providing timely feedback whether a given program is correct or not.
- Some student teams were very creative in presenting their results, and produced presentations far in excess of our initial expectations.
- Not all presentations are factually correct, and need immediate correction by the teachers as part of the assessment.

- A prepared sample solution available after the event can help correct misconception, and show the correct process for solving the problem. It also help to extend the time students are exposed to CP.
- In many cases, the students wrote statements in MiniZinc that were syntactically correct, but which did not represent the constraint they intended. A tool that can convert MiniZinc statements into natural language might be helpful, while not being too challenging to implement.
- While we use visualization throughout the course, and stress the importance of it in developing larger systems, we currently do not have much support to help students develop their own visualization as part of their assignment. While a switch to a Python-based constraint system might overcome this particular restriction, this would create new problems that we do not want to address at this point.
- Creating a new student assignment every year requires some non-trivial development effort (about two weeks each year), and is a challenge in itself: the problem needs to be understandable for students without any specific background, the scale of the problem should neither be too simple nor too complex, and multiple approaches to solving the problem should exist.
- One of the aims of the CRT-AI program is to help continued education inside industry, the current format of the training weeks is perhaps not optimized to achieve that goal.
- We consider the course format a success, mixing students from different universities in one course creates a better network of peers and advisors for the students, while helping to ensure that the course material is presented by experts in each specific domain. This approach really is only workable with a national program like CRT-AI, which does carry some of the costs associated.

References

1 Morten Mossige. CSPLib problem 073: Test scheduling problem. `http://www.csplib.org/Problems/prob073`.

2 Nicholas Nethercote, Peter J. Stuckey, Ralph Becket, Sebastian Brand, Gregory J. Duck, and Guido Tack. Minizinc: Towards a standard CP modelling language. In Christian Bessiere, editor, *Principles and Practice of Constraint Programming - CP 2007, 13th International Conference, CP 2007, Providence, RI, USA, September 23-27, 2007, Proceedings*, volume 4741 of *Lecture Notes in Computer Science*, pages 529–543. Springer, 2007. `doi:10.1007/978-3-540-74970-7\_38`.

3 Helmut Simonis. Lessons learned from developing an on-line constraint programming course. In *WTCP 2015 - Workshop on Teaching Constraint Programming*, 2015.

4 Helmut Simonis and Guido Tack. Visualization for constraint programming. Tutorial at CP 2021, `https://zenodo.org/record/5589114`, 2021.

A Example Student Problem: Berth Allocation Problem - Dublin Port (2023)

This section shows the problem statement presented to the students for the 2023 Constraint Week problem. The students must read and interpret the problem statement to decide which problem they have to solve, how to formulate the problem in MiniZinc, and how to check and present the results. While not based on an actual project, the problem is rather close to some real-world applications of Constraint Programming, while scaled to allow solving as a student project. Note that the problem is given to the students in increments. At the initial presentation, only the problem up to and including the Monday Section is available. On Tuesday and Wednesday, more details of the problem and further extensions are given.

> This document describes the problem for this year's Programming Challenge in the CRT-AI 2023 Constraint Week. The problem is a common issue for ports around the world: where do you berth (place) the ships arriving at your harbour so that they can be unloaded and loaded with minimal disruption and delay.

A.1 Introduction

Dublin port is the largest port in Ireland, handling 60% of the overall volume of freight arriving and departing Ireland by ship. The port handles a mix of different ship types:

1 - RoRo roll-on, roll-off ships carry vehicles which are driven on and off the ship. Typical examples are ferries and car carriers.
2 - LoLo lift-on, lift-off ships carry containers that are lifted by cranes on and off the ship.
3 - Bulk bulk carries transport products like ore, coal, or products like wheat, which fill the holds on the ship.
4 - Liquid Bulk liquid bulk carriers typically are oil tankers, which carry crude or processed oil in their holds, which can be pumped on and off the ship to special tank farms on land.
5 - Cruise cruise ships transport passengers only, and require special facilities for the boarding and disembarking of passengers

The overall layout of the port is shown in Figure 12, taken from the 2023 edition of `https://www.dublinport.ie/about-dublin-port/yearbook/`, the facilities occupy both sides of the mouth of the river Liffey.

Ships can be placed at specific locations called berths in the harbour which are defined by the type of ship that can be handled, the maximal length and depth of the ship that the berth can accommodate, and the location in the port. The numbering of the berths in Dublin port has evolved over time, with most unused numbers being further upriver. Table 1 lists the berths currently available in the harbour, we use a consecutive number to identify the berth, instead of the more irregular name given in the second column.

Dublin Port provides a list of all ship arriving and staying at the harbour on their website (`https://www.dublinport.ie/information-centre/ships-in-port/`), an example is shown in Figure 14. A more generic interface is provided by VesselFinder, which also shows the position of ships in and outside the port (Figure 13).

Our task is to write a program that, given a set of ship arrival and departure times, finds a compatible allocation of ships to berths in the harbour, so that every berth holds at most one ship at a time.

Figure 12 Layout of Dublin Port

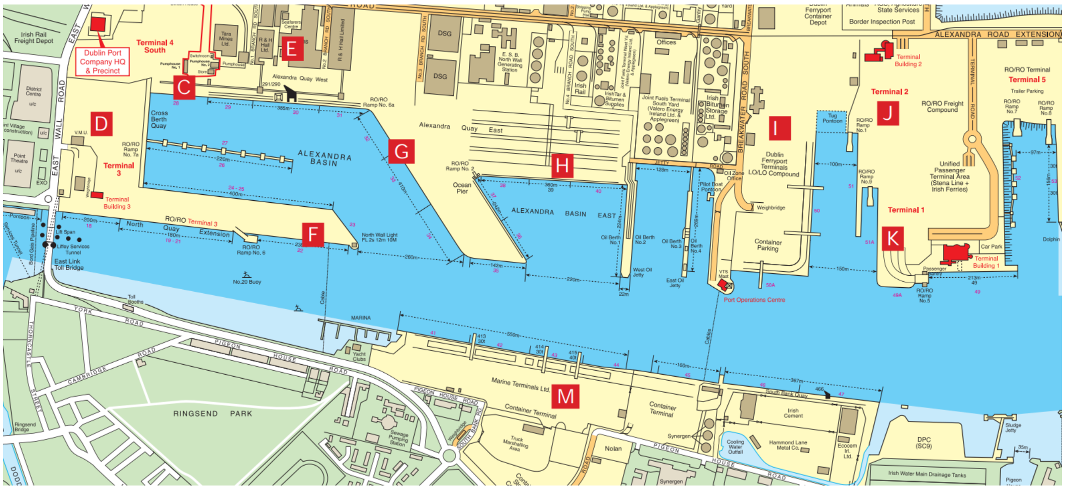


Figure 13 Ship Location Provided by VesselFinder

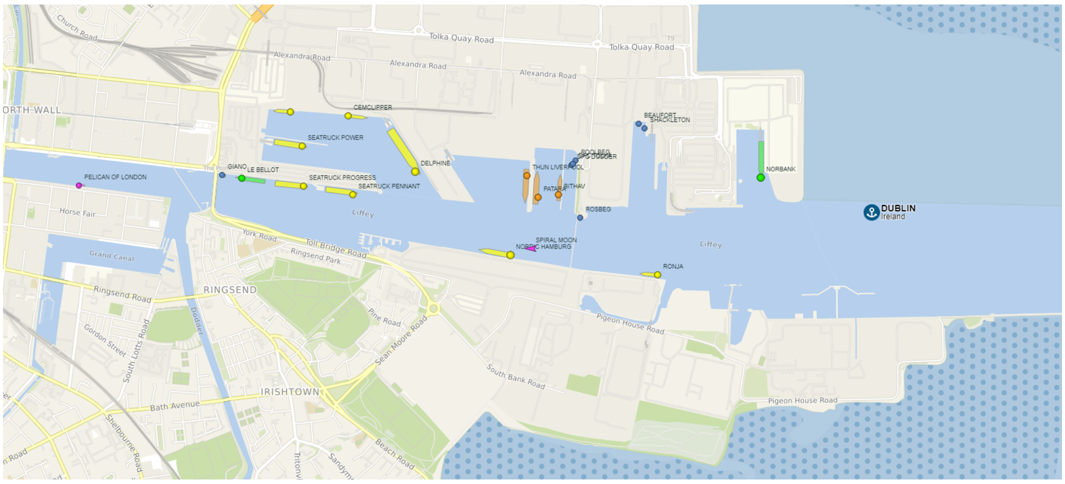


Table 1 Berths Available

Nr	Name	Type	Length (m)	Depth (m)
1	18	Cruise	200	6.5
2	26	RoRo	156	5.9
3	27	RoRo	156	5.9
4	28	Bulk	100	6.1
5	29	Bulk	125	9.0
6	30	Bulk	125	9.0
7	31	Bulk	125	9.0
8	32	LoLo/Bulk	140	9.5
9	33	LoLo/Bulk	140	9.5
10	34	Bulk	130	9.5
11	36	RoRo	120	10.3
12	37	RoRo	120	10.3
13	38	LoLo/Bulk	120	10.0
14	39	LoLo/Bulk	120	10.0
15	40	LoLo/Bulk	120	10.0
16	41	LoLo	120	7.4
17	42	LoLo	140	11.0
18	43	LoLo	140	11.0
19	44	LoLo	140	8.0
20	46	Bulk	180	11.0
21	47	Bulk	180	11.0
22	49	RoRo	213	11.0
23	50	LoLo	290	11.0
24	50a	LoLo	290	11.0
25	51	RoRo	205	8.0
26	51a	RoRo	190	8.0
27	52	RoRo	200	8.0
28	Oil 1	Liquid Bulk	220	11.0
29	Oil 2	Liquid Bulk	220	11.0
30	Oil 3	Liquid Bulk	220	11.0
31	Oil 4	Liquid Bulk	220	11.0

Figure 14 Dublin Port Information about Ships in Port

VESSEL NAME	VESSEL TYPE	ARRIVAL DATE	DEPARTURE TIME	BERTH
SHINGLE	BULK SOLID	24/06/2014 13:27	15/05/2023 23:58	OCEAN PIER 33
VOE VANGUARD (TUG)	WORK VESSEL	09/12/2017 22:47	10/12/2017 03:06	49
GIANO	WORK VESSEL	20/06/2022 21:06	31/12/2023 23:58	18
ROMI	BULK SOLID	04/05/2023 12:03	08/05/2023 23:58	DEEP WATER BERTH 46
SEATRUCK PENNANT	RORO FREIGHT/PASSENGER	06/05/2023 09:45	08/05/2023 14:00	21
SEATRUCK POWER	RORO FREIGHT/PASSENGER	06/05/2023 21:28	09/05/2023 03:30	26
PELICAN OF LONDON (TALL SHIP)	SAIL TRAINING VESSEL	07/05/2023 13:45	08/05/2023 11:00	SJR QUAY 8
SAMSKIP ENDEAVOUR	LOLO	07/05/2023 15:23	08/05/2023 23:45	50N
SOLONG	LOLO	07/05/2023 19:40	08/05/2023 15:30	ALEX BASIN EAST 39
JSP ROVER	LOLO	07/05/2023 22:05	09/05/2023 13:00	MTL 42
AN HAI WAN	BULK LIQUID	07/05/2023 23:53	08/05/2023 23:58	OIL BERTH NO. 3
ELBTRADER	LOLO	08/05/2023 00:32	09/05/2023 15:30	50S
PARITY	BULK SOLID	08/05/2023 01:35	11/05/2023 22:00	ALEX QUAY WEST 30
NORBAY	RORO FREIGHT/PASSENGER	08/05/2023 05:40	08/05/2023 21:15	53
ARKLOW BANK	BULK SOLID	08/05/2023 05:52	09/05/2023 18:00	DEEP WATER BERTH 47
SEATRUCK PROGRESS	RORO FREIGHT/PASSENGER	08/05/2023 06:11	08/05/2023 20:00	26
LAURELINE	RORO FREIGHT/PASSENGER	08/05/2023 08:37	08/05/2023 20:00	OCEAN PIER 33
THUN LUNDY	BULK LIQUID	08/05/2023 09:05	09/05/2023 23:59	OIL BERTH NO. 1
LAGERTHA	BULK LIQUID	08/05/2023 09:11	12/05/2023 23:58	OIL BERTH NO. 2

A.2 Monday

We are given some data files, which describe the harbour and the planned ship movements. The data use the following MiniZinc format

```
enum ShipType;
int:nrBerths;
array[1..nrBerths] of set of ShipType:berthType;
array[1..nrBerths] of int:berthLength; % in meters
array[1..nrBerths] of int:depth; % in centimeters

int:nrShips;
array[1..nrShips] of ShipType:shipType;
array[1..nrShips] of int:shipLength;  % in meters
array[1..nrShips] of int:draught; % in centimeters
array[1..nrShips] of int:arrival; % integer, 15min time resolution
array[1..nrShips] of int:departure; % integer, 15 min time resolution
```

The arrival time is given as an integer, measured in 15 minutes units since the start of the week. Note that some berths accept more than one type of ship, the berthType is a set of enumerated ShipType values, while each ship has a specific ShipType value.

The datafiles are `monday1.dzn`, `monday2.dzn`, and `monday3.dzn`.

An example datafile is shown below:

```
ShipType={RoRo,LoLo,Bulk,Liquid,Cruise};

nrBerths=5;
berthType =[{RoRo},{Bulk,LoLo},{RoRo},{Bulk},{Cruise}];
berthLength=[100,150,100,100,100];
depth=[800,1100,900,900,700];

nrShips=7;

shipType=[RoRo,RoRo,Bulk,RoRo,Cruise,LoLo,LoLo];
shipLength=[90,90,100,100,100,150,150];
draught = [800,800,800,800,600,1100,1100];
arrival=[0,0,0,100,100,0,100];
departure=[50,100,150,220,250,50,150];
```

A.2.1 Tasks

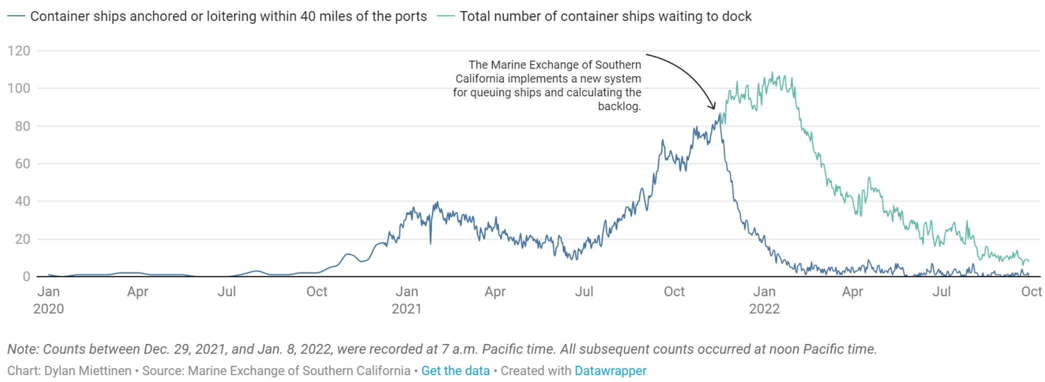
1. For each of the data files, find an assignment of ships to berths so that each ship is in a compatible berth, and no two ships occupy the same berth at the same time (that is considered to be a bad thing). A ship is compatible with a berth if it has a compatible type, the ship is smaller than the length of the berth, and the draught of the ship is less than the depth of water in the berth.
2. What makes a good assignment, which assignment is preferable to another one?
3. Can you explain what is happening in dataset 3?

A.3 Tuesday

Sometimes, it is not possible to assign a free berth to each ship immediately as it arrives at the harbour, and a ship may have to wait outside the port until a berth becomes available. We should update our solution in such a way that we may assign a delayed arrival time for a ship, after the originally planned arrival time given in the data. In the case of a delayed arrival we also need to shift the departure time, as the time needed for unloading and loading does not change with the delay. Usually we want to treat all ships fairly, so that a ship that is planned to arrive before another ship will not be in fact berthed after that later ship's berthing, but this fairness rule only applies to ships to the same type. Some ships or ship types may get priority treatment, we are more likely to delay a bulk freighter than a passenger liner, and capacity problems for LoLo ships should not affect the RoRo ferries.

Shipping delays due to port congestion can be very serious. During 2021, up to a hundred ships were waiting to enter the Port of Los Angeles in the United States (Figure 15). This was largely due to limited container storage capacity at the port, but also an effect of reduced manpower availability due to Covid restrictions.

Figure 15 Backlog at the Port of LosAngeles, USA in 2021



A.3.1 Tasks

1. Modify the program of Monday to deal with the delayed berth assignment. A ship can only be assigned an arrival time after the original planned arrival.
2. How do you express the fairness rule within each ship type?
3. A natural objective is to minimize the sum of all delays imposed on all ships. How does the fairness rule affect this objective?

A.4 Wednesday

A.4.1 Tugs

In order to safely navigate in the approach to the harbour, larger ships need a tug to help them with the movement in and out of the harbour. In Dublin Port, two tugs, called Beaufort and Shackleton, are on duty 24/7 (See Figure 16, image from `https://afloat.ie/port-news/dublin-port/item/52722-dublin-port-tug-pair-take-up-new-home-within-the-port-estate`). The role of the tug is explained in this video `https://www.dublinport.ie/videogallery/tug-boats-action-large-ships-enter-dublin-port/`.

Figure 16 Dublin Port Tugs

For our model, we assume that every ship that is longer than 150m will need a tug for 30 minutes (2 time units) before arrival, and after departure, and that cruise ships require both tugs for their movement in and out of the harbour.

A.4.2 Tides

The tides affect how much water is available in the channel leading into the harbour, and this affects at which times vessels with deep draught can enter the port. Table 2 from `https://www.dublinport.ie/information-centre/tide-tables/` shows the tides in Dublin port for this week.

For our model, we can assume that ships with a draught of more than 10 m can only enter or depart the port three hours before and after high tide. This restricts the arrival and departure times for these large ships.

Table 2 Tide Table for Dublin Port

DATE	TIME	TIDE HEIGHT
2023-05-14	01:05:00	1.4
2023-05-14	07:44:00	3.6
2023-05-14	13:53:00	0.9
2023-05-14	20:46:00	3.5
2023-05-15	02:16:00	1.3
2023-05-15	08:54:00	3.8
2023-05-15	14:57:00	0.8
2023-05-15	21:48:00	3.6
2023-05-16	03:17:00	1.1
2023-05-16	09:54:00	3.9
2023-05-16	15:51:00	0.6
2023-05-16	22:41:00	3.7
2023-05-17	04:09:00	0.9
2023-05-17	10:48:00	4
2023-05-17	16:39:00	0.6
2023-05-17	23:27:00	3.8
2023-05-18	04:55:00	0.8
2023-05-18	11:36:00	4
2023-05-18	17:22:00	0.6
2023-05-19	00:06:00	3.8

A.4.3 Tasks

1. Modify your model to consider the impact of using tugs and respecting tide constraints for ships. How much additional delay do these two constraint cause?
2. Is it worthwhile to consider a third tug for the harbour? How would you estimate the impact of another tug over a longer period of time?
3. To allow large vessels to enter the port without tidal restrictions, one can dredge the channel for greater depth. How would you use the constraint model to estimate the value of such an operation?