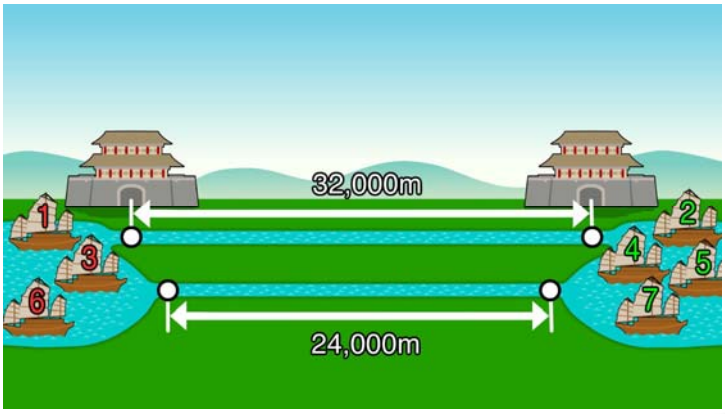
Sequence Dependent Scheduling 2

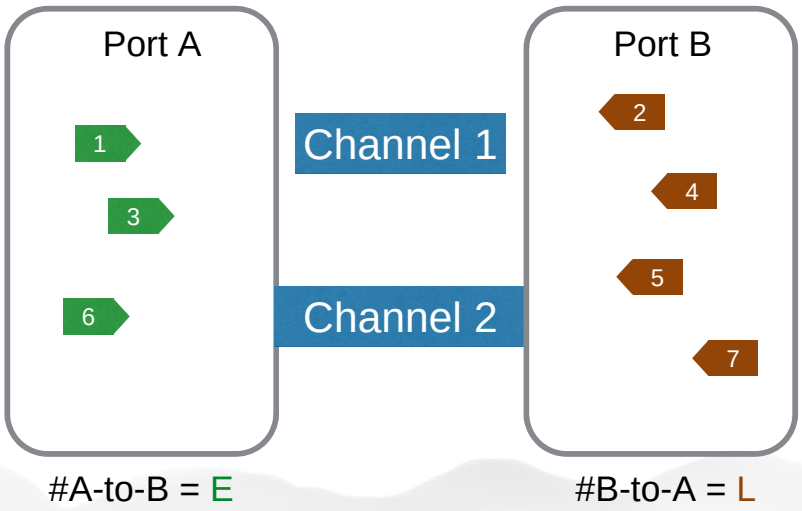
Jimmy Lee & Peter Stuckey

One More Channel



2

Double Channel Problem



3

Double Channel Problem

- Given ports *A* and *B* connected by channels 1 and 2. Consider a set of *E* ships going from port *A* to *B*, and *L* ships going from *B* to *A*. We need to choose (a) which channel to use for each ship and (b) when the ships should enter the selected channels
  - Each ship has a specific speed and can leave no earlier than a desired time for that ship
  - Channels 1 and 2 are 24000m and 32000m long
  - A ship can enter only if the channel is **clear**, i.e. no ships sail in opposite directions simultaneously
  - Two ships cannot be closer than 2000m
  - Minimize the time to move all the ships

4

Double Channel (doubleChannel.mzn)

- Data

```
int: nC; % number of channels
array[1..nC] of int: len;

int: nS; % number of ships
set of int: SHIP = 1..nS;
array[SHIP] of int: speed;   % 1000m time
array[SHIP] of int: desired; % desired time
int: atob = 1; int: btoa = 2;
array[SHIP] of atob..btoa: dirn;

int: leeway;  % leeway between 2 ships
int: maxt;        % maximum time
set of int: TIME = 0..maxt;
```

5

Double Channel (doubleChannel.mzn)

- Decisions
  - add **nC** dummy ship**s** as last ships in each channel
  - so each ship will have a next ship in its channel

```
set of int: SHIPE = 1..nS+nC; % add dummies
int: dummy = 3;
array[SHIPE] of atob..dummy: kind
           = dirn ++ [dummy | i in 1..nC];
array[SHIPE] of int: speede =
   speed++[0|i in 1..nC];
array[SHIPE] of var TIME: start;
array[SHIPE] of var TIME: end;
array[SHIPE] of var 1..nC: channel;

array[SHIP] of var SHIPE: next; % next ship
```

6

Double Channel (doubleChannel.mzn)

- Constraints
  - dummy ships are last **and in a fixed channel**

```
forall(s in nS + 1 .. nS + nC)
      (start[s] = maxt /\ end[s] = maxt);
forall(s in nS + 1 .. nS + nC)
      (channel[s] = s - nS);
```

- Relationship between start and end

```
forall(s in SHIP)(end[s] = start[s] +
                  len[channel[s]]*speed[s]);
```

- The next ships are all different

```
alldifferent(next);
```


Double Channel (doubleChannel.mzn)

- Relationship between a ship and its next ship
  - the start and end time are constrained

```
forall(s in SHIP)
          % ships of opposite dirn
      (if kind[s] + kind[next[s]] = 3 then
          end[s] <= start[next[s]]
       else % same dirn
start[s]+speed[s]*leeway <= start[next[s]] /\
  end[s]+speede[next[s]]*leeway <= end[next[s]]
       endif);
```

  - **they are in the same channel**

```
forall(s in SHIP)
      (channel[next[s]] = channel[s]);
```

Double Channel (doubleChannel.mzn)

- Cannot leave before desired time

```
forall(s in SHIP)(start[s] >= desired[s]);
```

- Objective

```
solve minimize max(s in SHIP)(end[s]);
```

9

Solving the Model

```
start = [55, 43, 47, 33, 115, 125, 175, 400,
400];
end = [175, 235, 143, 193, 243, 293, 271, 400,
400];
channel = [1, 2, 1, 2, 2, 1, 2, 1, 2];
next = [6, 5, 1, 2, 7, 8, 9];
----------
==========
```

- Down from almost 22 days to slightly over 12 days

10

Order Dependent Setup Times

- If there are order dependent setup times or costs
  - model the next task explicitly
  - add constraints to ensure the setup time or cost is paid
- Examples are:
  - direction change in channel
  - mode change in machine shop scheduling
  - cool down time for smelting jobs at different temperatures

11

Summary

- Complex scheduling applications
  - have interdependencies between a task and the task that follows it
- We need to model for each task
  - which task is next, and
  - how that constrains the schedule

12

Image Credits

All graphics by Marti Wong, ©The Chinese University of Hong Kong and the University of Melbourne 2016