UNIVERSITY OF Southampton
School of Mathematics

How to load a ferry: a comparison of packing algorithms for the vehicle ferry industry

C. Bayliss, A. Martinez-Sykora, C. Currie, M. So, J.A. Bennell

ICCL Southampton October 2017

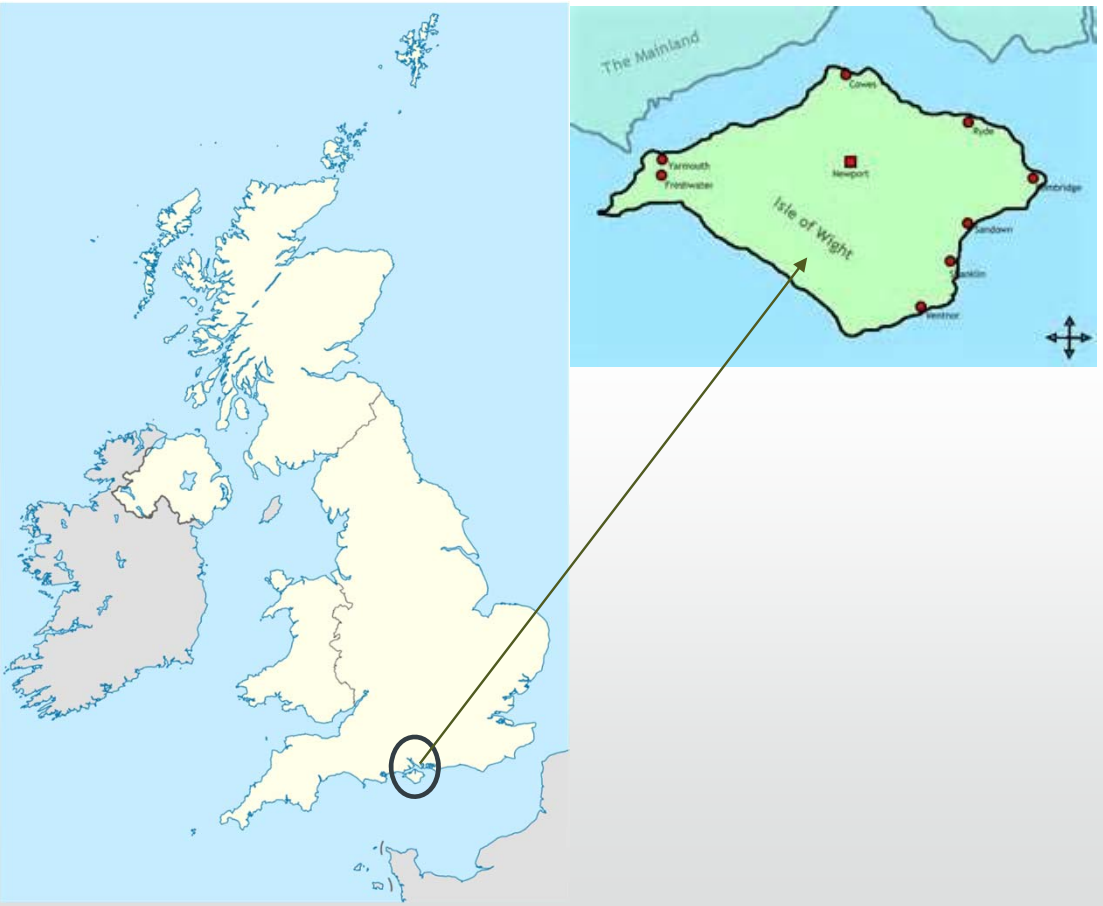
This work was funded by the EPSRC under grant number EP/N006461/1

UNIVERSITY OF Southampton
School of Mathematics

Talk overview

1. Description of the vehicle ferry loading problem
2. Previous work: Two packing algorithms
    1. Application: Optimal dynamic pricing
3. An algorithm for queue constrained loading
    i. Application: Minimise penalties for failing to load vehicles
4. Conclusion

Case study



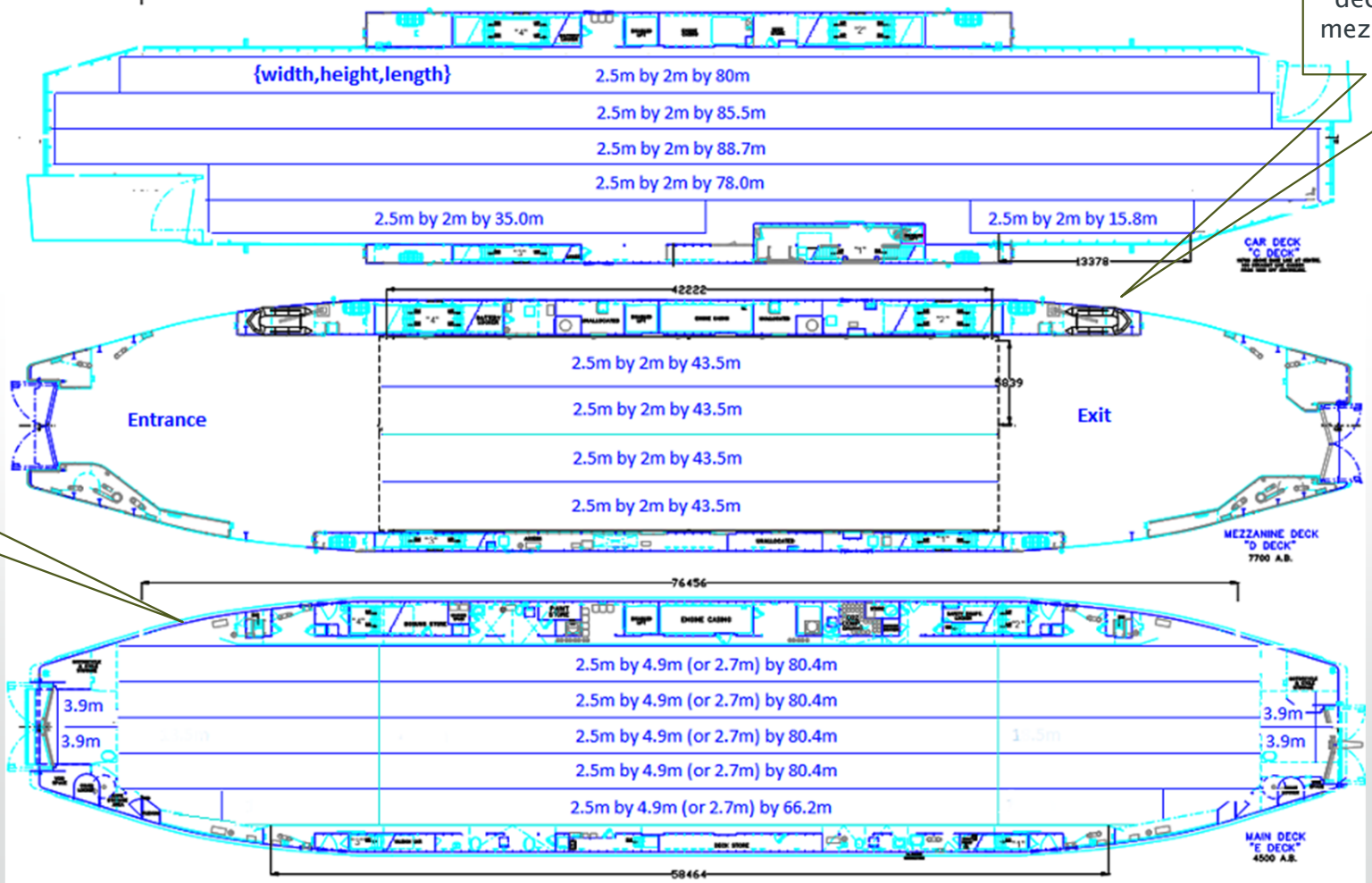
- Red Funnel: regular crossings between Southampton and the Isle of Wight
- Accommodates private vehicles and commercial freight vehicles

UNIVERSITY OF Southampton
School of Mathematics

Ferry dimensions

Lane parking of cars and motorcycles on the upper deck and cars only on the mezzanine decks when they are in operation

Multiple vehicle types are parked on the main deck. 2D packing Is allowed



UNIVERSITY OF Southampton
School of Mathematics

- Southampton terminal
- Parallel queues for pre-sorting vehicles

UNIVERSITY OF Southampton
School of Mathematics

Description of the vehicle ferry loading problem

- Vehicles who have purchased tickets **arrive at random times** before departure
- Upon arrival vehicles are **directed to terminal queues** according to their type and dimensions
- Some **vehicles are prioritised** in the loading process such as those which:
  - require unimpeded access to the lifts
  - hold a priority boarding pass
  - drop trailers
- Minimum **parking gaps** have to be respected so that passengers can exit the vehicle deck for the duration of the journey

UNIVERSITY OF Southampton
School of Mathematics

Problem description

- Some vehicles have **large turning circles** which means that it is best to avoid parking them in the corner positions either side of the exit
- If **mezzanine decks** are used some parts of the main deck will have lower maximum vehicle height constraints
- Some parking positions require a **reverse gap** to be left so that vehicles are not jammed on the ferry at exit time

UNIVERSITY OF Southampton
School of Mathematics

One-dimensional bin packing (1DBP)
Two-dimension packing heuristic (2DH)

TWO PACKING ALGORITHMS

UNIVERSITY OF Southampton
School of Mathematics

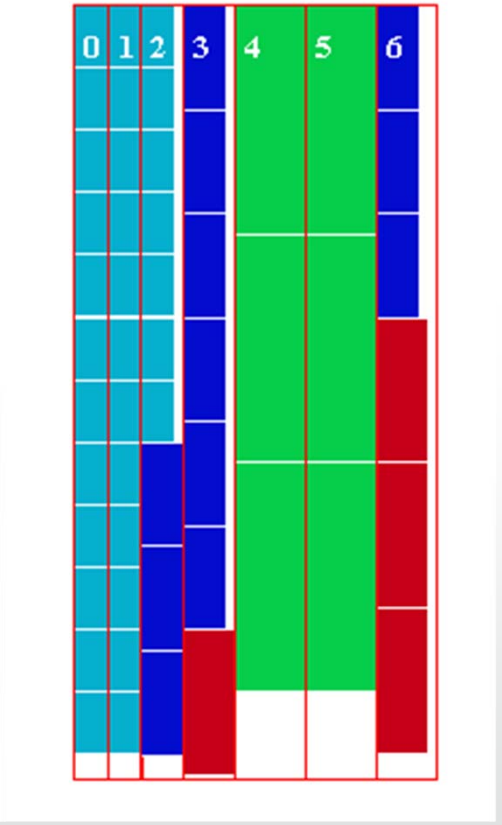
One-dimensional bin packing (1DBP)

- Each deck consists of a set of lanes (bins)
- Each bin has a width, height and length (constraints)
- Maximise the value of the packed vehicles
- $Max \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} \sum_{k=1}^{|J_j|} s_{ijk} x_{ijk}$

$$\sum_{j \in J} \sum_{k \in J_j} x_{ijk} \leq d_i \quad \forall i \in I$$

$$\sum_{i \in I} l_i x_{ijk} \leq \hat{l}_j \quad \forall j \in J, \qquad \forall k \in J_j$$

$$x_{ijk} \in \mathbb{N} \quad \forall \{\forall i \in I, \qquad \forall j \in J, \qquad \forall k \in J_j\}$$



UNIVERSITY OF Southampton
School of Mathematics

One-dimensional bin packing (1DBP)

- Extended to allow:
  - for vehicles **parked across adjacent lanes**
  - for **variable lane sizes**

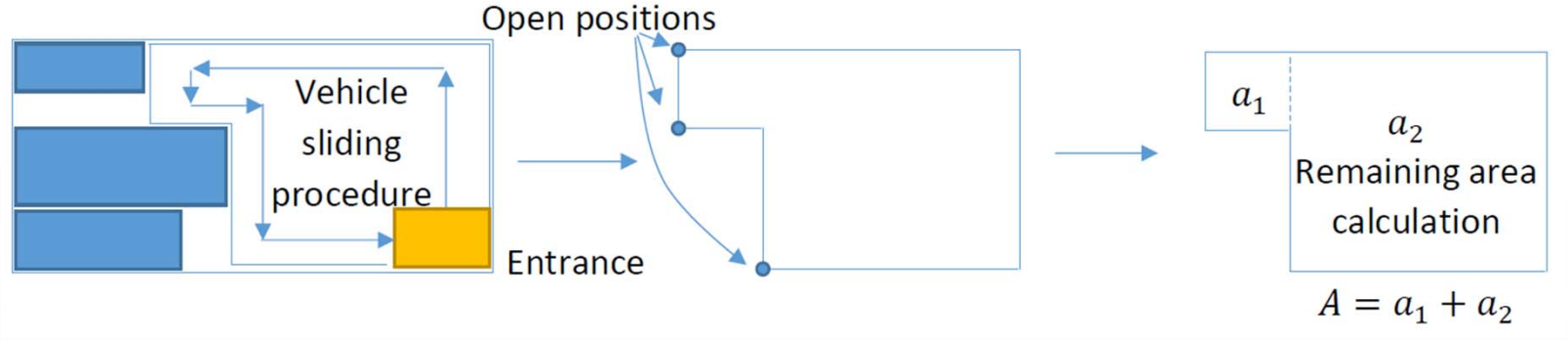
UNIVERSITY OF Southampton
School of Mathematics

Two-dimensional packing heuristic (2DH)

- This heuristic is designed for use in a **loading simulator** developed in collaboration with Red Funnel and accounts the real world constraints
  - Large vehicle manoeuvrability; lift access; lowerable mezzanine decks; drop trailers and also reverse gaps
- The loading simulator follows the **procedure implemented in the real world**
  1. Load cars and motorcycles onto the car deck and mezzanine decks if used
  2. Priority vehicle types are then loaded onto the main deck
  3. The remaining vehicles are loaded onto the main deck using the two-dimensional packing heuristic

Two-dimensional packing heuristic (2DH)

- **A vehicle sliding procedure** was used to track the remaining vehicle positions and to calculate efficiency attributes for those positions



- Loading decisions where chosen as those which maximised a weighted sum of efficiency attribute scores, simulated annealing was used to set the attribute weights to maximise packing efficiency
  - $\max_j \sum_{i=1}^{n} w_i a_{ji}$

UNIVERSITY OF Southampton
School of Mathematics

Optimal dynamic pricing problem

Objective: derive a dynamic pricing policy that maximises the expected revenue from the sale of vehicle tickets on a ferry

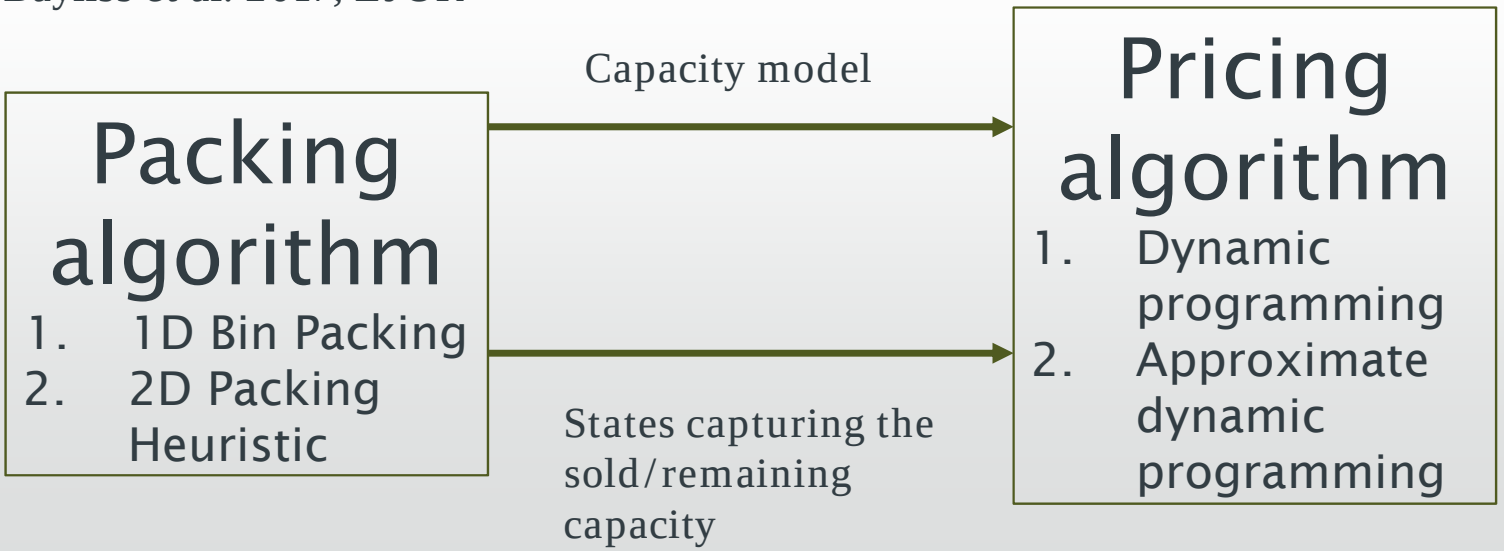
- **Constraint:** Limited **capacity** which **depends on packing efficiency**
- Customers
  - Arrive at random times during the **selling season** (beginning 6 months before departure)
  - Have a maximum **willingness-to-pay** is dependent on time until departure and their vehicle type
  - And, their vehicles vary in **size**

UNIVERSITY OF Southampton
School of Mathematics

General framework for integrating optimal packing and pricing

- The **optimal dynamic pricing policy** can be calculated using dynamic programming
- The states capture remaining capacity based on **optimal packing** of the accepted vehicles

- Submitted:
  - 1. Martinez-Sykora et al. 2017, Trans Res B
  - 2. Bayliss et al. 2017, EJOR



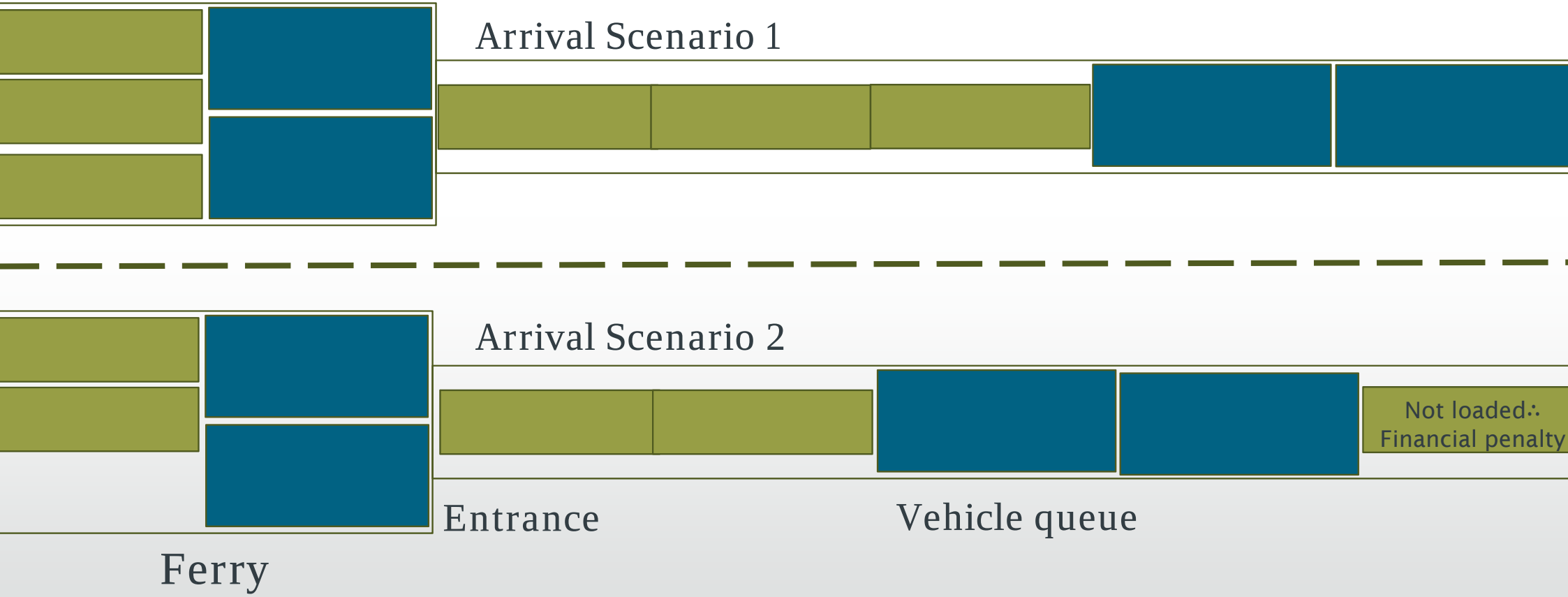
UNIVERSITY OF Southampton
School of Mathematics

Sequential Guillotine Cut Knapsack (SGCKS)

QUEUE CONSTRAINED VEHICLE FERRY LOADING

UNIVERSITY OF Southampton
School of Mathematics

Queue constraints can make a packing problem infeasible



This problem can occur whenever it is not possible to keep all vehicle types in separate queues

UNIVERSITY OF Southampton
School of Mathematics

Sequential **G**uillotine-**C**ut-**K**nap**S**ack packing approach (**SGCKS**)

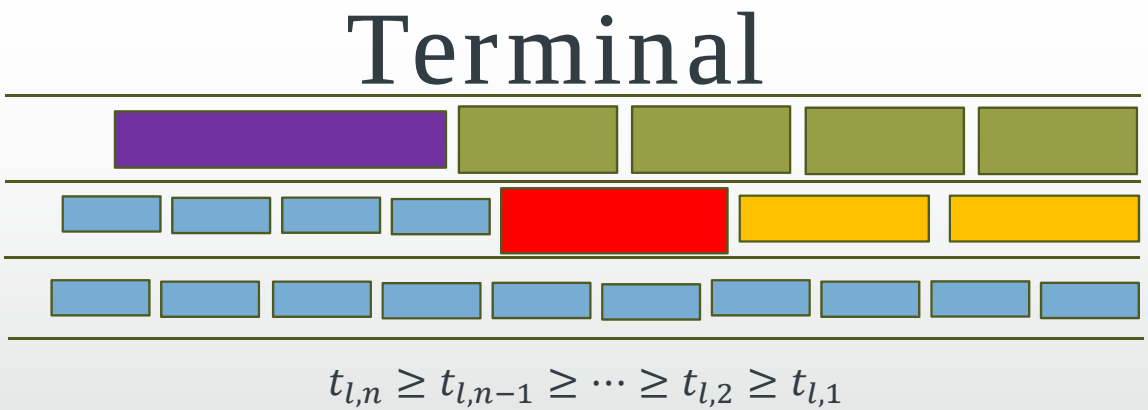
- **Implementable** and intuitive
- The yard (queueing) policy and packing solutions are each encoded as **integer strings**
- In both cases integers encode **size** and **orientation** information
- **Queues** are built by applying the yard policy during the vehicle arrival process
- **Packing solutions** are constructed by performing a sequence of horizontal and vertical cuts to the remaining ferry space, each cut-off space is packed with vehicles to create a **row or a column of vehicles**
- Packing solutions obey **queue order constraints** as at each stage vehicles can only be loaded if they are currently at the front of a queue

UNIVERSITY OF Southampton
School of Mathematics

Yard policy solution encoding

	Quantiles			
Strip type		small	middle	large
	Width	0	1	2
	Length	3	4	5

Example solution={2,5,3}



2=Vehicles with a large width

5=Vehicles with a large length

3=Vehicles with a small length

$$t_{l,n} \geq t_{l,n-1} \geq \cdots \geq t_{l,2} \geq t_{l,1}$$

The queue orders are dependent upon random arrival times and a lane allocation policy

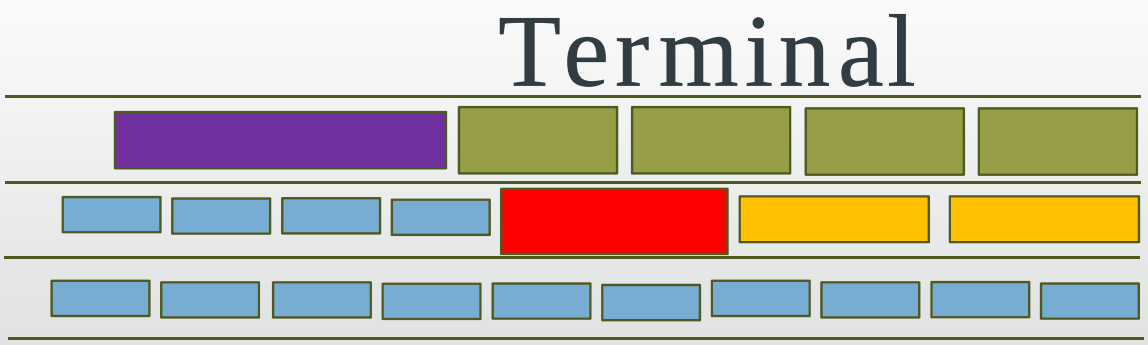
UNIVERSITY OF Southampton
School of Mathematics

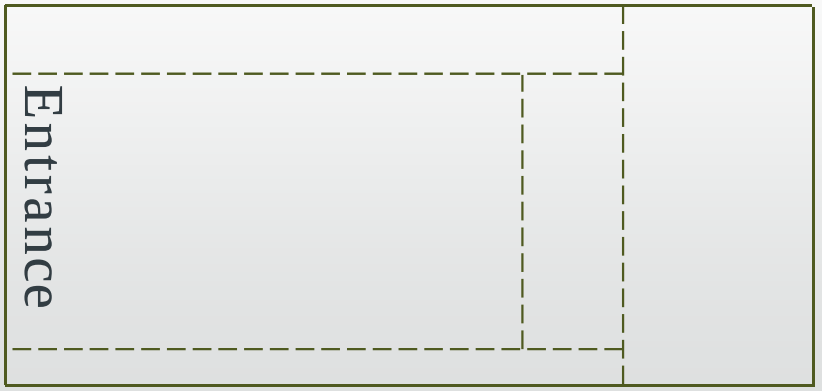
SGCKS packing solution encoding

	Quantiles			
Strip type		small	middle	large
	Bottom row	0	3	6
	Left column	1	4	7
	Right column	2	5	8

- 3 cut orientations
- 3 rectangle size quantiles

Example solution={3,2,7,0,6}

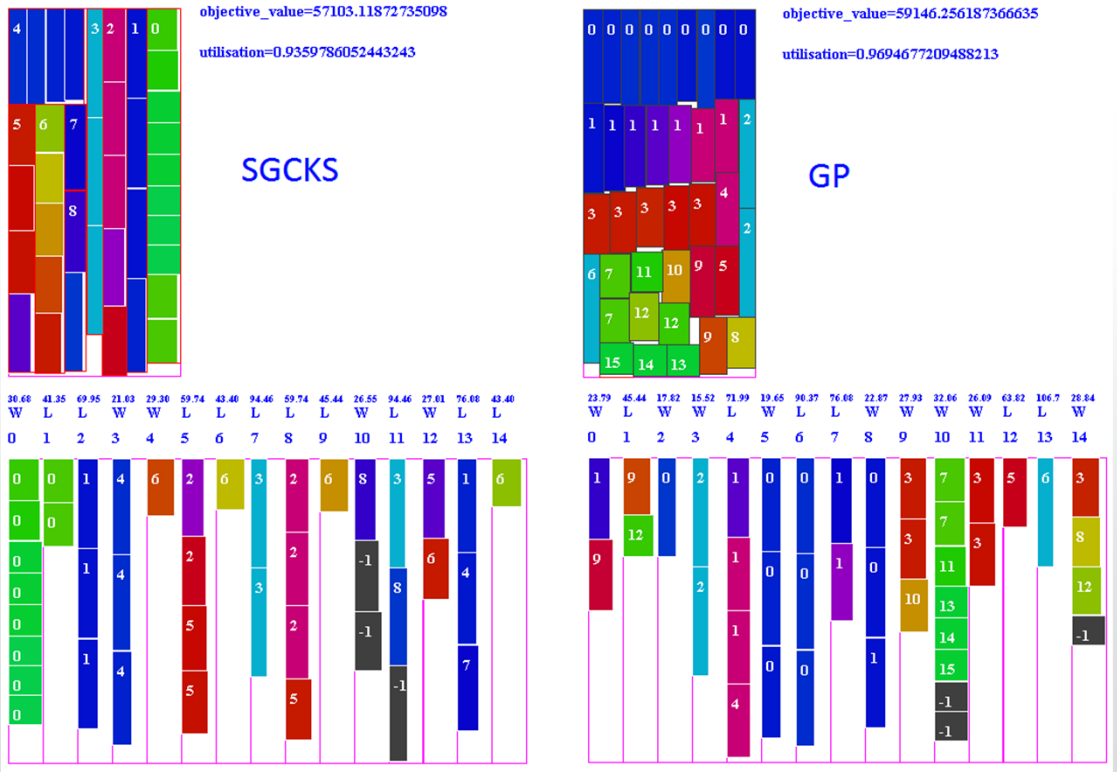




UNIVERSITY OF Southampton
School of Mathematics

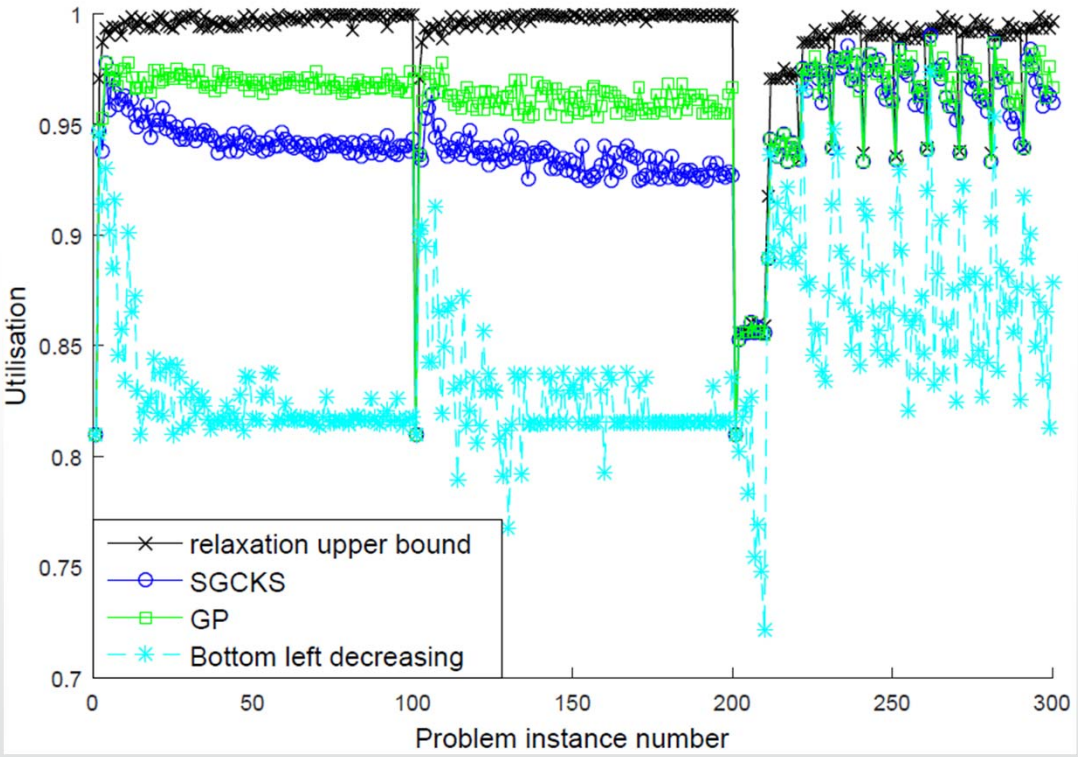
SGCKS relaxation (General Packing or GP)

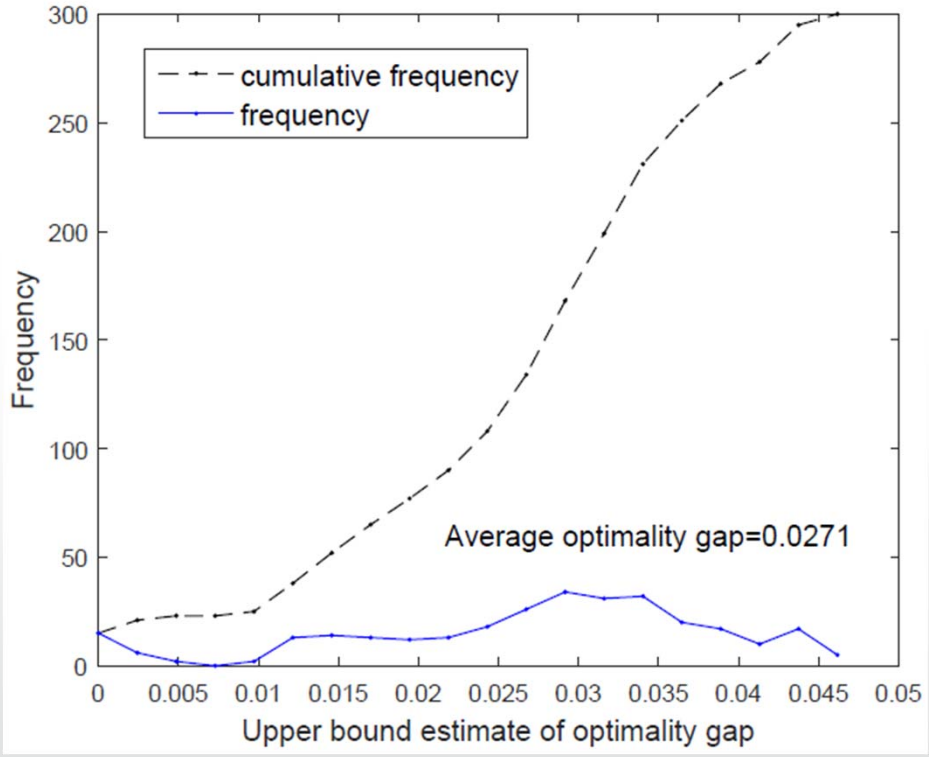
- Instead of strictly obeying the vertical and horizontal cuts use the nearest corner positions, this recovers wasted space within previous cuts



UNIVERSITY OF Southampton
School of Mathematics

Utilisations and optimality gaps for SGCKS approaches





UNIVERSITY OF Southampton
School of Mathematics

Queue constrained packing revenue problem

Objective: maximise revenue from the sold vehicle tickets minus penalties for failing to load any vehicles

- **Penalty: refund times two (refund plus compensation)**
- **Decisions:**
  - **Yard policy** for arrival vehicles
  - **Commit to loading a subset of vehicles**, as to rule out compensation payments to those we refund pre-emptively
- Expected revenue=Committed vehicle revenue-2(revenue of non-loaded vehicles)

UNIVERSITY OF Southampton
School of Mathematics

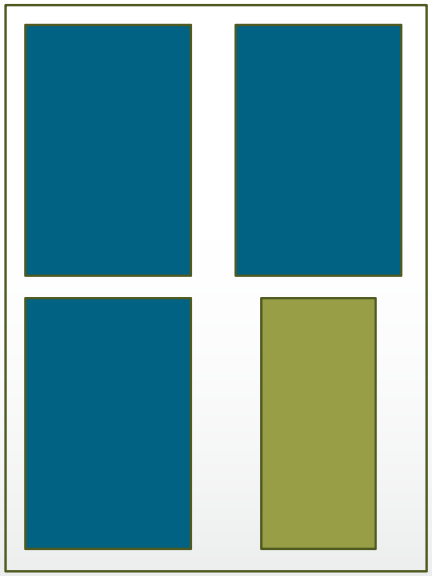
Solution strategy

- Given a set of **random arrival scenarios** S
- We propose finding a yard policy $\boldsymbol{y}$ and a set of packing solutions $\boldsymbol{P}$, one for each arrival scenario such that the **revenue value of the intersection vehicle mix is maximised**
- The intersection vehicle mix is that which we **commit to loading**
- A **risk parameter** $\boldsymbol{w}$ is introduced, we then maximise the revenue value of the intersection of w out of $|S|$ vehicles mixes

UNIVERSITY OF Southampton
School of Mathematics

Maximise the intersection vehicle mix

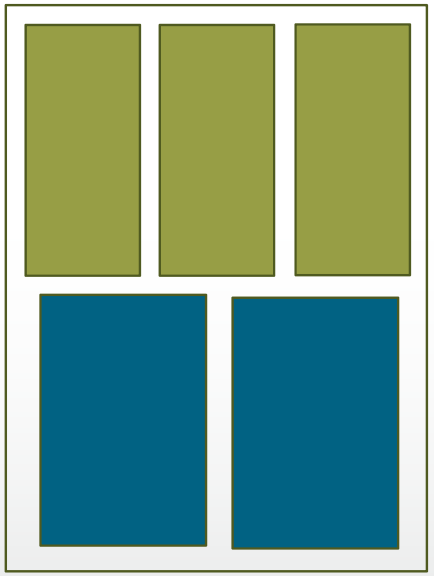
Arrival scenario 1 packing solution



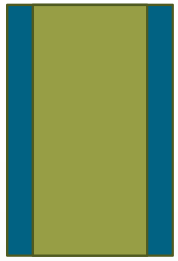
2 small vehicles left off

Tickets sold={3,3}

Arrival scenario 2 packing solution



Nested vehicle size



1 large vehicle left off

{1,2}: Simple intersection (just take the minimum of each vehicle type)
{2,2}: Intersection after accounting for nested vehicle sizes

Given the possible arrival scenarios we should commit to loading {2,2}

UNIVERSITY OF Southampton
School of Mathematics

Formulation

- $\max_{y,P} \sum_{j \in J} R_j m_j$ -Revenue value of committed vehicle mix
- Subject to

1. $m = \bigcap_{b \in B}^{*} v_b$ -m is the intersection (committed) vehicle mix
2. $B \subset S$ -over a subset of the arrival scenarios
3. $|B| = w$ -the size of the subset (risk parameter)
4. $v_s \leftarrow g\big(p_s, f(y,s)\big) \quad \forall s \in S$ -the SGCKS methodology maps y, P to loaded vehicle mixes

UNIVERSITY OF Southampton
School of Mathematics

Iterative solution approach

- An iterative metaheuristic alternates between packing and yard policy optimisation
- Fix one optimise the other in each iteration
  - Packing iterations evaluate the impact of forcing particular arrival scenarios and their candidate packing solutions as elements of B
  - Yard iterations evaluate candidate yard policies in the effect on the revenue value of the intersection* vehicle mix
- Due to $v_s \leftarrow g\big(p_s, f(y,s)\big)$ of SGCKS searching y also searches P

The effect of |B| and |S|

Revenue		Subset size (B)														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Uncertainty	1	55330														
set	2	56240	56778													
size	3	55781	56352	56921												
(S)	4	55214	57029	56680	55262											
	5	55494	56607	56827	56595	54862										
	6	55373	56767	56851	**57188**	56129	54721									
	7	55682	56731	57021	55847	55667	55940	54776								
	8	56225	56404	56524	56504	56548	54828	54825	54252							
	9	55516	56700	57148	56933	56277	55411	54828	53878	52969						
	10	55918	56289	56343	56980	56482	56080	54756	53807	55349	54014					
	11	55992	56779	56849	56925	56377	56206	55543	54554	54557	54414	52922				
	12	55909	55892	57075	56919	56891	54781	56172	54915	53882	54889	53272	52820			
	13	54907	56373	56769	56728	56157	55482	56015	56173	54727	54154	53622	52169	52448		
	14	56092	56052	56562	56993	56236	56396	54889	55169	53342	54600	54171	53668	54411	52105	
	15	55392	56118	56205	56980	56832	55514	56236	55900	55572	54672	53312	52191	53878	54460	53652

- For this fully nested 30 vehicle category example revenue was maximised by setting |S| and |B| as high as possible
- When there is no nested vehicle type structure moderate |S| and |B| values prevent us from competing over uncommon vehicle mixes

UNIVERSITY OF Southampton
School of Mathematics

Comparison of packing algorithms

Constraint/feature	1DBP	2DH	SGCKS	Sim-SGCKS
Parking gaps	0.5	1	0.5	1
Priority vehicles	0	1	0	1
Height restrictions	1	1	1	1
Reverse gaps	0	1	0	1
Mezzanine decks	0	1	0	1
Turning circles	0	1	0	1
Irregular ferry shape	1	1	1	1
Random arrival/yard policy	0	0	1	WIP
Queue orders respected	0	0	1	WIP
Two-dimensional packing	0.5	1	1	1
Optimised	1	0.5	0.5	0.5

UNIVERSITY OF Southampton
School of Mathematics

Conclusion

- Our research project in collaboration with the Red Funnel vehicle ferry company has given rise to **three different packing models**
- The initial 1DBP formulation formed the basis of an analytical formulation of **optimal dynamic pricing integrated with optimal packing**
- The 2DH method applied the integrated pricing and packing framework in a **larger scale** and **2D packing** setting with heuristics and approximate dynamic programming methods
- The SGCKS methods were developed to account for the effects of random arrivals and **queue constrained packing**
- Implementing the SGCKS approach within the loading simulator brought everything together in a **training tool for loading personnel**

UNIVERSITY OF Southampton
School of Mathematics

SGCKS viewed as functions

- The SGCKS packing methodology can be viewed as follows:
  - Queueing process:
    - $q \leftarrow f(y, s)$ that is a yard policy y and an arrival scenario s map to a set of vehicle queues q
  - Packing process:
    - $v \leftarrow g(p, q)$ this a SGCKS packing solution string p and a set of vehicle queues q maps to a vector of the loaded vehicle types of each type

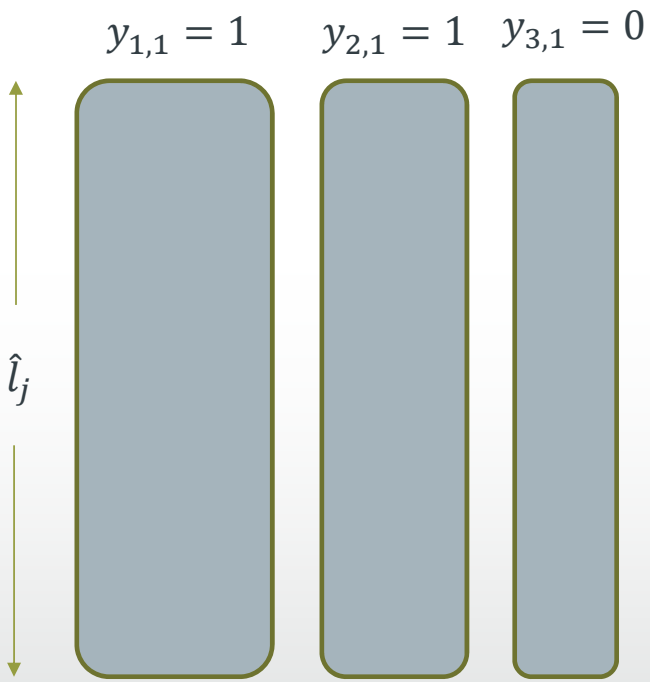
UNIVERSITY OF Southampton
School of Mathematics

Three packing algorithms overview

- One-dimensional bin packing
  - Lane parking model, each lane is a bin
  - Can be optimised efficiently via Integer Programming techniques
- Two-dimension packing heuristic
  - Simulation of case study with real world constraints
  - Parking decisions determined by weighted sum of efficiency attribute measurements, weights optimised by simulated annealing
- Sequential Guillotine Cut Knapsack
  - Guillotine cuts create horizontal/vertical bins
  - The bins are packed in cut order whilst respecting queue order constraints

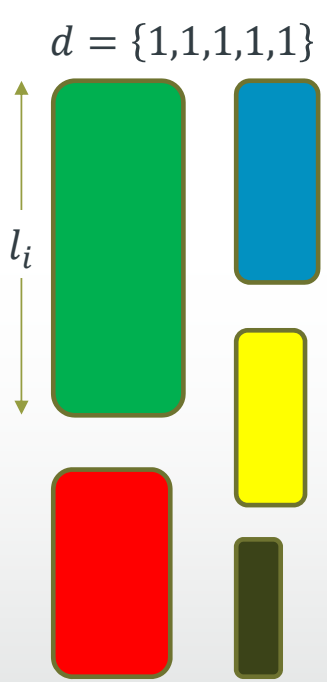
UNIVERSITY OF Southampton
School of Mathematics

Variable lanes 1DBP formulation (Exact)



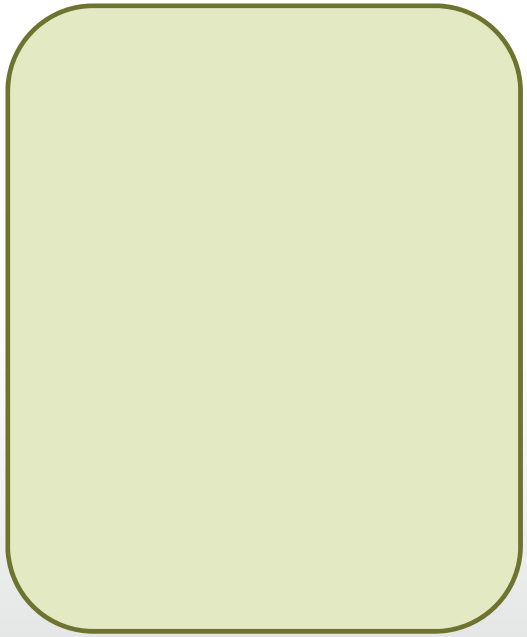
Set of lane types (bins)

$j \in J$



Set of vehicle types

$i \in I$



Ferry

$x_{1,1,1} = 1$

$x_{2,1,1} = 1$

$x_{3,2,1} = 1$

$x_{4,2,1} = 1$

$x_{5,2,1} = 1$

UNIVERSITY OF Southampton
School of Mathematics

Lower bounds for utilisation

- LB given by the area of the rectangles/bin – Usually very poor
- Simple DFF provide usually poor LB
- We propose to solve two 1DBPPs to improve the lower bound given by the area of the rectangles.

Steps:
1- Solve 1dbpp (horizontally) – LB1
2- Solve 1dbpp (vertically) – LB2
3- LB=LB1+LB2-Interection
4- Iteratively reduce the number of vehicles of each type to update LB

UNIVERSITY OF Southampton
School of Mathematics

Lower bounds for utilisation – **1DBPP**

Minimum waste we generate horizontally

Master Problem:

$$Min \sum_{j \in J} w_j x_j$$

Waste of pattern j

$$\sum_{j \in J} d_j x_j \geq D_j$$

Where x_j represents the number of times pattern j is used in the solution

Sub problem:

$$Min \sum_{i \in I} (a_i + l_i) y_i$$

$$\sum_{i \in I} l_i y_i \leq L$$

Where y_i represents the number of times piece i is used in the pattern, a_i the dual reduced costs, l_i the length of rectangle i and L the length of the bin.

UNIVERSITY OF Southampton
School of Mathematics

Lower bounds for utilisation – **1DBPP**

We look into the original pieces

w_j - real waste in the 2D problem

We repeat the same procedure with the width of the pieces and the bin – LB2

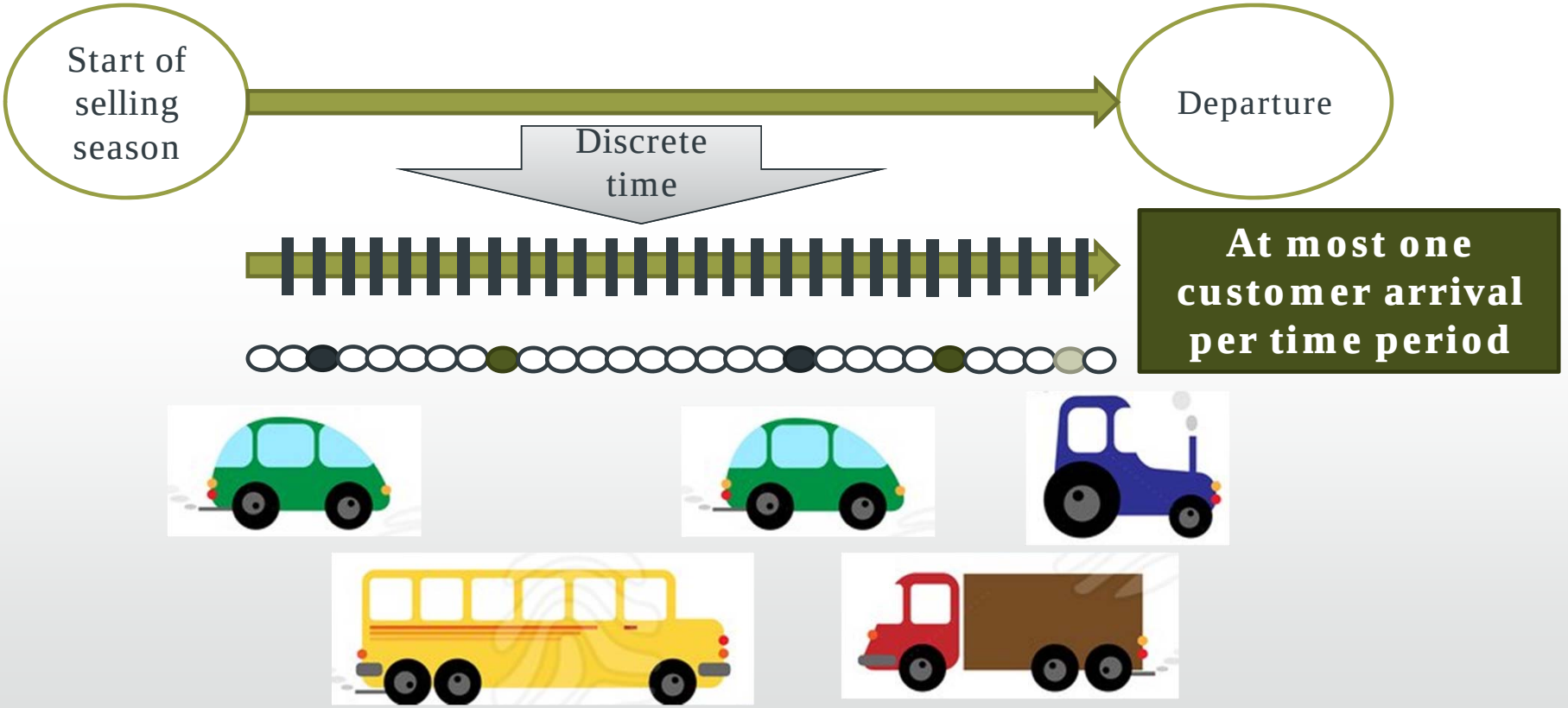
UNIVERSITY OF Southampton
School of Mathematics

Demo

- Sim-SGCKS
  - The SGCKS-GP method implemented within the loading simulator (but currently without yard policy WIP)

UNIVERSITY OF Southampton
School of Mathematics

Selling Tickets



UNIVERSITY OF Southampton
School of Mathematics

Optimising the loading rules

- The **loading algorithm** sequentially selects where it would load each vehicle type and then selects which vehicle type to load next
- For each **Loading decision** a number of efficiency based attributes are calculated
- Example attributes:
  - Distance from the exit of the ferry (bottom)
  - Distance from the side of the ferry (left)
  - Tightness (vehicle width/parking position width)
  - Parking loss (space lost due to staggered parking (gaps))
  - Bottom overlap adjacency ratio (mouth full)

UNIVERSITY OF Southampton
School of Mathematics

Optimising the loading rules (continued)

- Loading decisions (d) are **scored by a weighted sum** of the attribute measurements and the decision with the highest score is selected
  - Loading decision $= \max_{d \in decisions} \left(\sum_{i=1}^{attributes} w_i a_i^d \right)$
- The simulated annealing algorithm **optimises the weights w_i** to achieve efficient loading
- The **fitness function** used by the simulated annealing algorithm is:
  - Max: $objVal =$ (on-line remaining space) $-M_1$(number of wasted gaps) $-M_2$(area of vehicles that do not fit)

UNIVERSITY OF Southampton
School of Mathematics

Applications of each algorithm

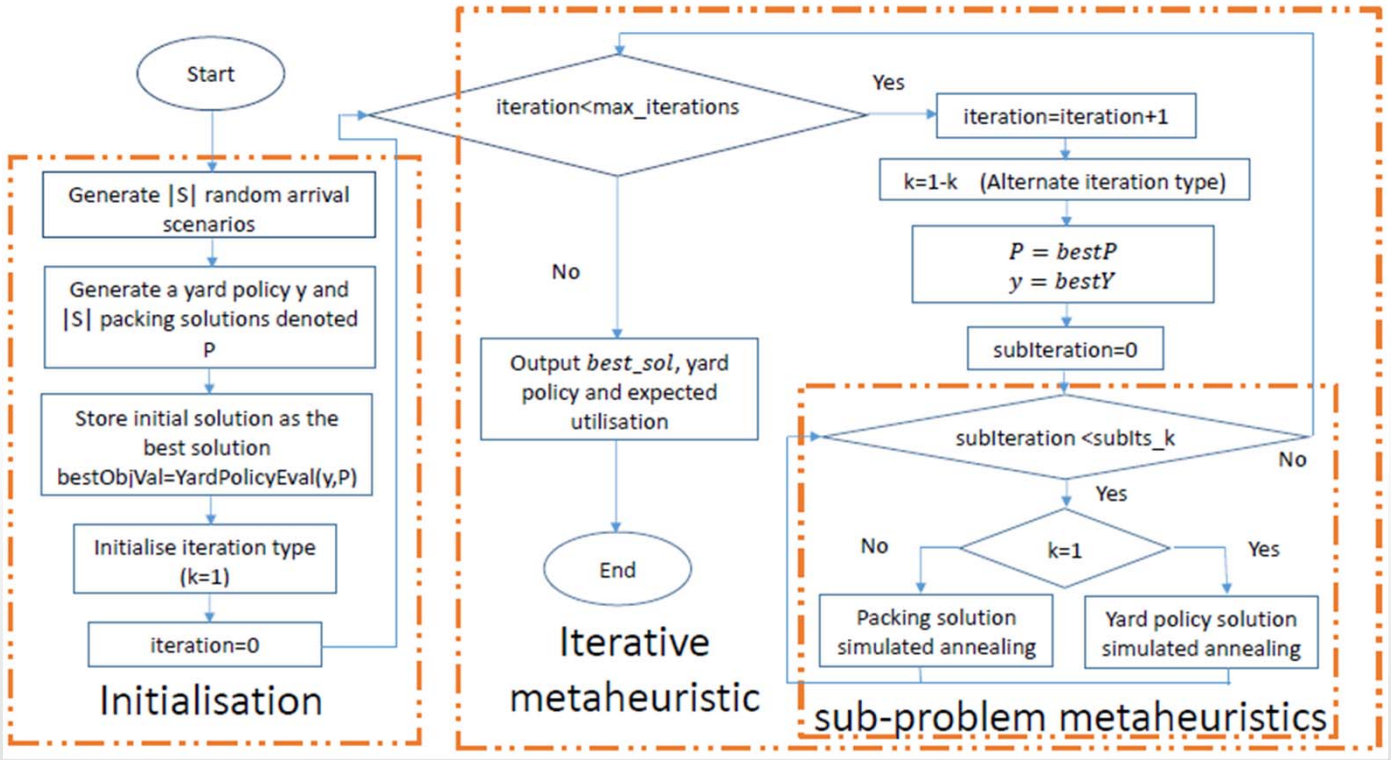
- **1DBP**: used to identify all possible packing solutions as states in a dynamic programming formulation of **optimal dynamic pricing** in the vehicle ferry industry. Martinez-Sykora et al. 2017 submitted TRANS RES Part B.
- **2DH**: used to implement a 2D packing formulation of the **dynamic pricing** formulation for **larger ferries and more vehicle types** by using the simulator to map vehicle mix states to lower dimension remaining space states. Bayliss et al. 2017 submitted EJOR.
- **SGCKS**: used in method to determine vehicle mixes that can be loaded reliably under vehicle arrival time uncertainty and queue constrained packing. Bayliss et al. ESICUP 2017.
- **Sim-SGCKS**: Red Funnel hosted a workshop demonstration of this implemented as a **training tool** for loading personnel, the tool is being developed further based on their feedback.

UNIVERSITY OF Southampton
School of Mathematics

Results comparing the packing algorithms

- We compared 1DBP and 2DH dynamic pricing approaches in 1D packing framework and demonstrated the potential value of full 2D packing solutions
- We compared the SGCKS based approached with upper bounds derived from a relaxed packing formulation and with results from the bottom-left heuristic

Iterative metaheuristic algorithm



UNIVERSITY OF Southampton
School of Mathematics

Queue constrained packing under arrival uncertainty

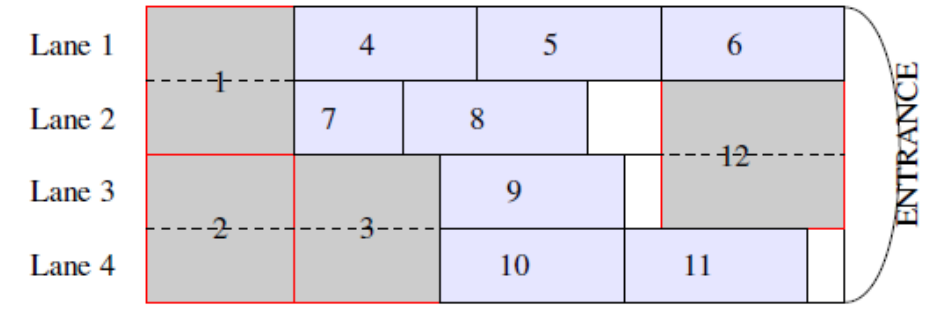
- S: set of arrival scenarios
- S: an arrival scenario is defined as
- m_v: number of vehicles of type v to commit to loading
- B: subset of arrival scenarios
- Y: yard policy
- P: set of packing solution strings

UNIVERSITY OF Southampton
School of Mathematics

Merged lanes formulation

- Sometimes wide vehicles have to be parked across two lanes, requires the following constraints

- $\sum_{j\in J}\left(\sum_{k\in J_j} x_{i,j,k} + \sum_{k'\in J'_j} z_{i,j,k'}\right) \le d_i \quad \forall i \in I$
- $My_{j,k} \ge x_{i,j,k} \quad \forall i \in I, \ \forall j \in J, \ \forall k \in J_j \backslash J'_j$
- $My_{j,k} \ge x_{i,j',k'} + z_{i,j,k} \quad \forall i \in I, \ \forall j \in J, \ \forall k \in J'_j$
- $\sum_{i\in I} l_i x_{i,j,k} \le \hat{l}_j \quad \forall j \in J, \ \forall k \in J_j$
- $\sum_{i\in I} l_i \left(x_{i,j',k'} + z_{i,j,k}\right) \le \hat{l}_j \quad \forall j \in J, \ \forall k \in J_j$



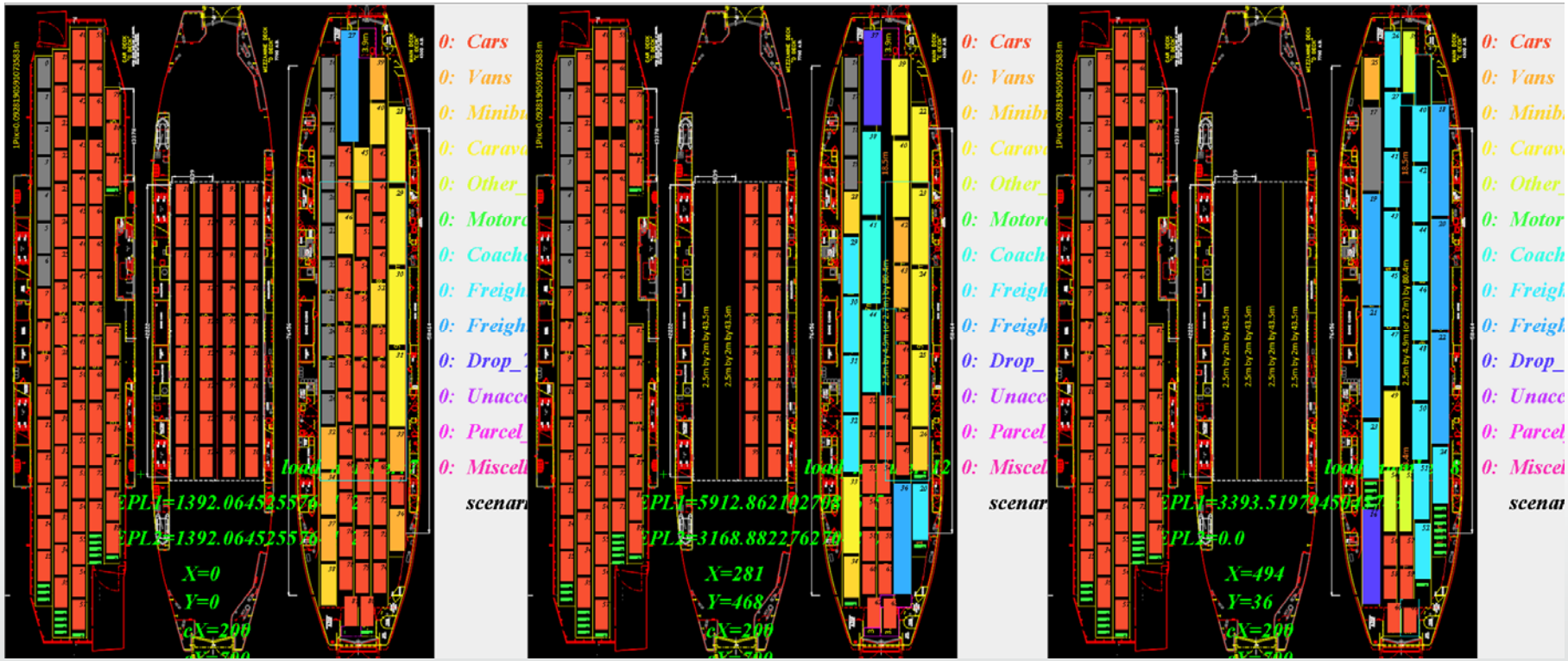
UNIVERSITY OF Southampton
School of Mathematics

Sim-SGCKS

- A slide on this

UNIVERSITY OF Southampton
School of Mathematics

Deck configurations and demand scenarios



High car demand
2 Mezzanine decks

Medium demand
1 Mezzanine deck

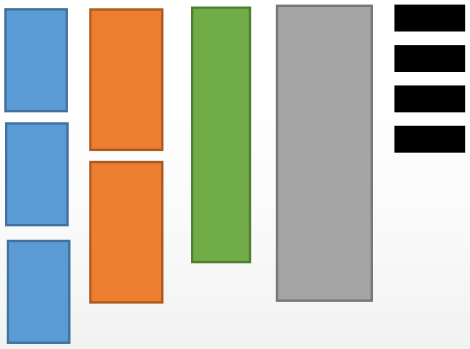
High freight demand
0 Mezzanine decks

UNIVERSITY OF Southampton
School of Mathematics

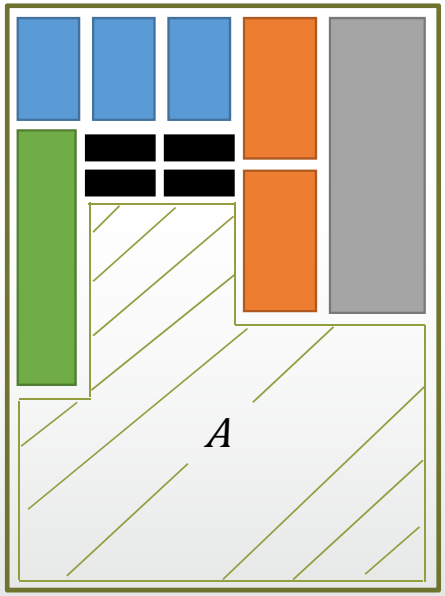
1DBP and 2DH selling season state definitions

1DBP

- 1-d bin packing model (optimal lane parking)
- State=count of vehicles of each type
- 1-dimension per vehicle type
- **Tickets sold state definition**



2DH



- 2-d packing heuristic (ignores lanes)
- State=remaining deck area per deck region
- 1-dimension per deck region
- **Remaining space state definition**

Transition functions for one vehicle type 0 sale

$state = 3,2,1,1,4$

$f(\{3,2,1,1,4\}, 0) = \{3,2,1,1,4\} + \{1,0,0,0,0\} = \{4,2,1,1,4\}$

$state = A = 950.8, \quad transition\ value = 26.2$

$f(950.8, 0) = 950.8 - 26.2 = 924.6$

UNIVERSITY OF Southampton
School of Mathematics

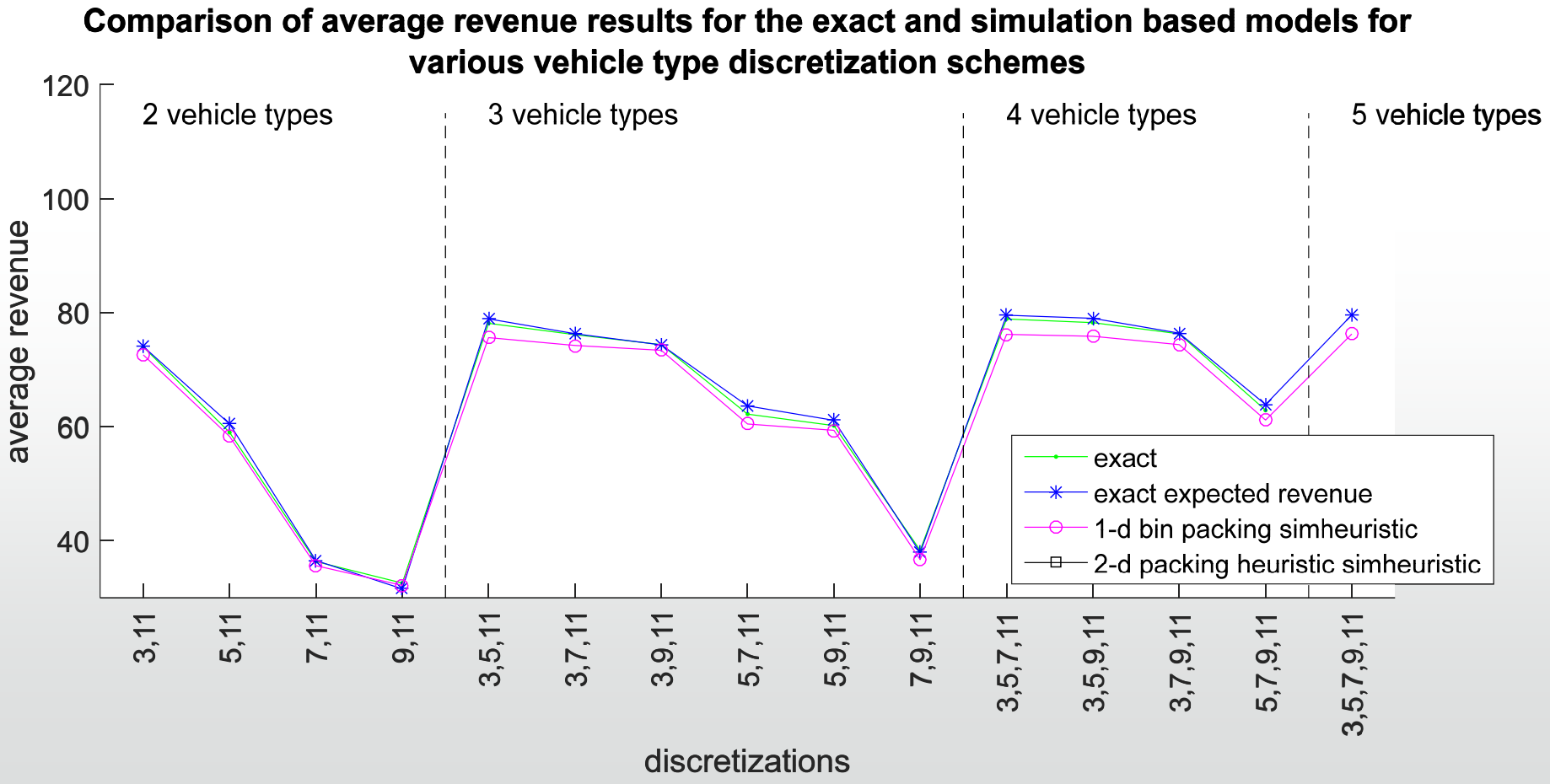
General framework for integrating packing and pricing

- Dynamic pricing formulation
  - The optimal dynamic pricing look-up-table policy can be derived by computing the Bellman equations by backwards recursion
  - In each state at each time 3 events can occur
    1. No customers arrive
    2. A customer arrives but does not purchase a ticket
    3. A customer arrives and purchases a ticket

$$V_t(s) = \max_{p \in P} \left\{ \left\{ \sum_{i \in I} \lambda_{t,i} \left\{ \alpha(i,p,t)\left(p + V_{t-1}\left(f(s,i)\right)\right) + \left(1 - \alpha(i,p,t)\right) V_{t-1}(s) \right\}^{(3)} \right\}^{(2)} + \lambda_{t,0} V_{t-1}(s) \right\}^{(1)}$$

UNIVERSITY OF Southampton
School of Mathematics

1DBP vs 2DH and the Benefit of 2-d packing



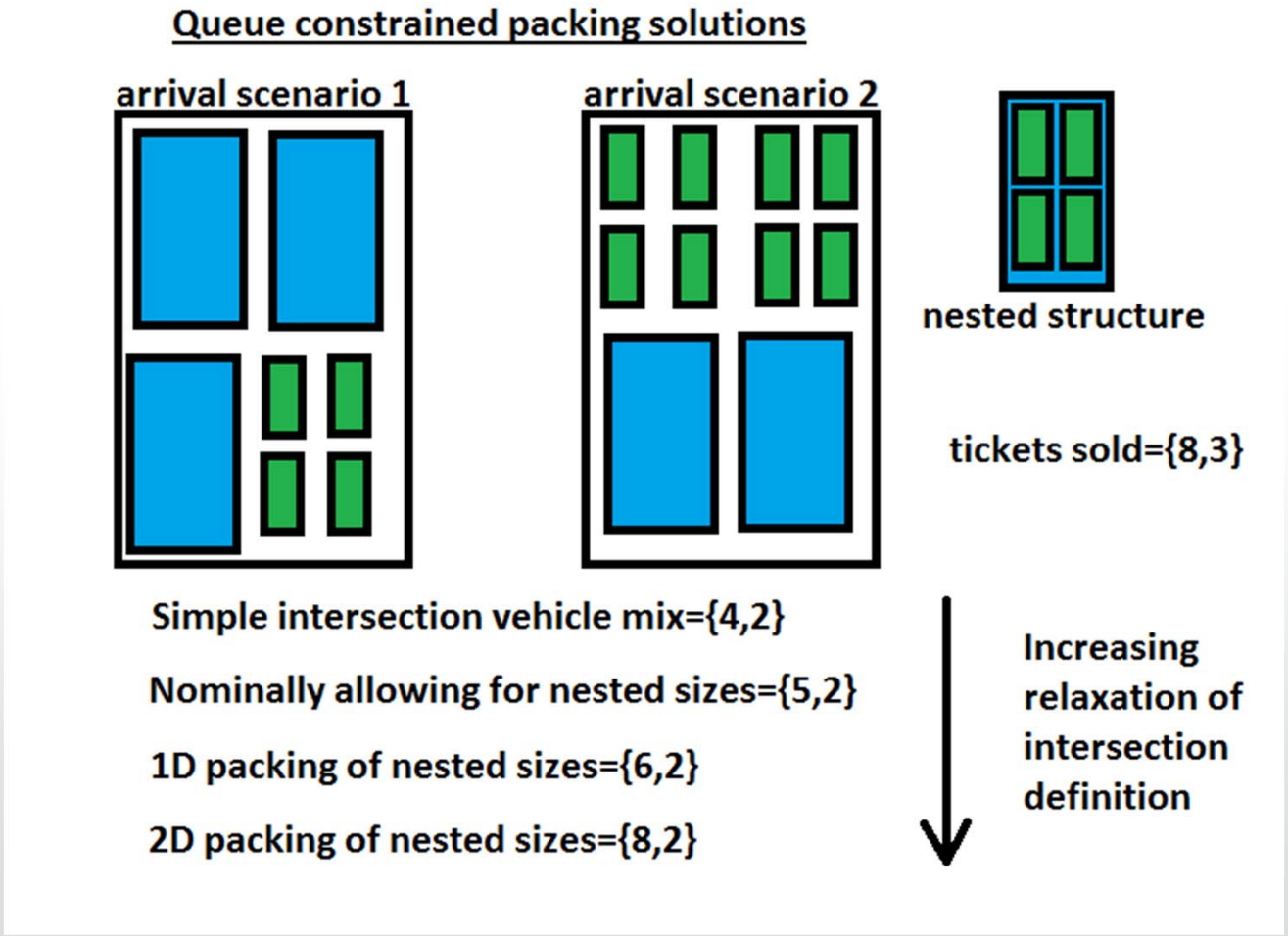
UNIVERSITY OF Southampton
School of Mathematics

Two decision problems

i. Optimal dynamic pricing in the vehicle ferry industry

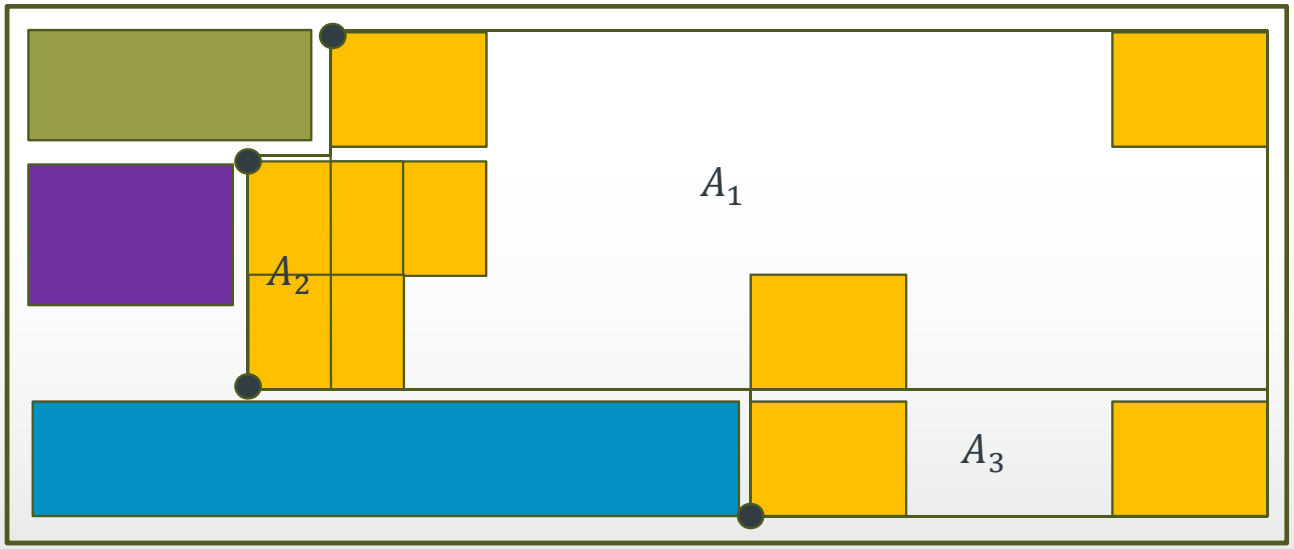
ii. Queue constrained packing under arrival uncertainty

Intersection of vehicle mixes



UNIVERSITY OF Southampton
School of Mathematics

On-line remaining space and parking position calculation



Available parking positions

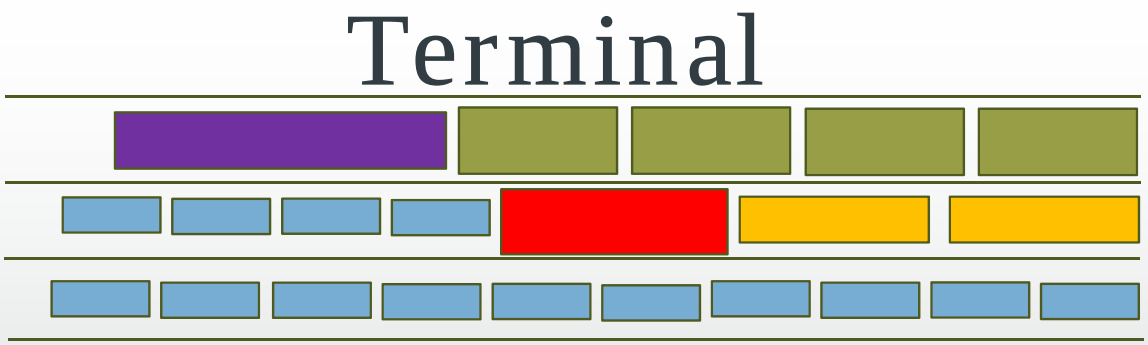
Remaining area which is used to map vehicle mix state to remaining space state

UNIVERSITY OF Southampton
School of Mathematics

Yard policy solution encoding

	Quantiles			
Strip type		small	middle	large
	Width	0	1	2
	Length	3	4	5

Example solution={2,5,3}



$$t_{l,n} \geq t_{l,n-1} \geq \cdots \geq t_{l,2} \geq t_{l,1}$$

Yard policy

- Each lane is devoted to vehicles of a target width or a target length
- On arrival vehicles are allocated to the closest matching lane
- Ties can be broken based on:
  - queue length
  - number of different vehicle types in queues

2=Vehicles with a large width

5=Vehicles with a large length

3=Vehicles with a small length

The queue orders are dependent upon random arrival times and a lane allocation policy