ICLR 2023 Presentation

Learning to CROSS-exchange to solve min-max vehicle routing problems

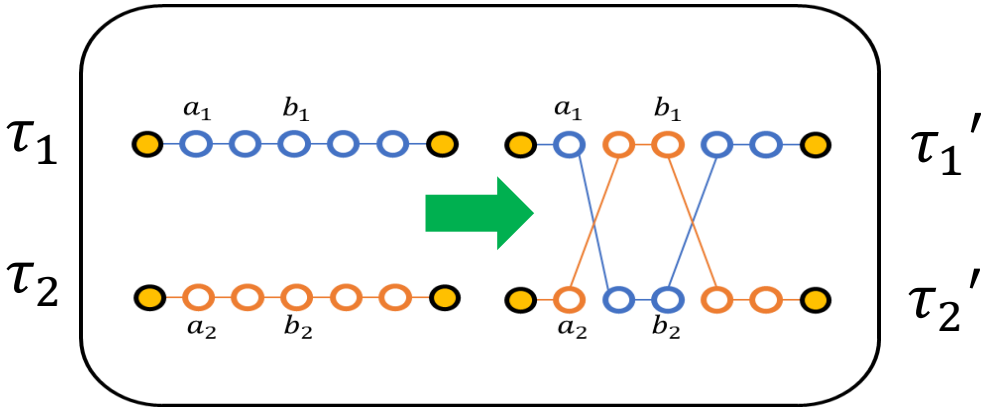
Dept. of Industrial & Systems engineering, KAIST
Minjun Kim, Junyoung Park, Jinkyoo Park

Motivation

- Current problem of Neuro Combinatorial Optimization (NCO)
  - Lack of practicality
    - NCO extensively focused on improving the performance of the benchmark CO problems
  - Lack of generalizability
    - NCO solver only can solve specific class or size of the problem
  - Lack of trainability
    - To train NCO solver, the solution of VRP instance is required

- Neuro CROSS exchange (NCE)
  - Focus on **min-max** routing problem
    - Lots of practical problems that aim to minimize the total completion time, for example, various **time-critical distributed tasks** (e.g., vaccine delivery, grocery delivery).
  - Learn core operator of heuristic
  - Learn cost-decrement model

Preliminaries: CROSS-exchange (CE)

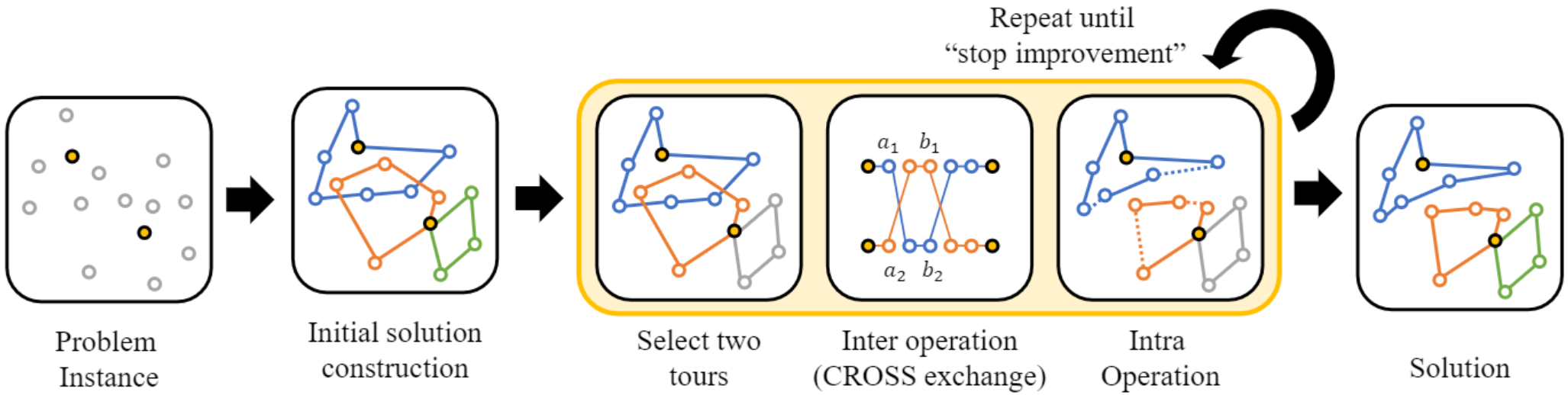


$$\tau_1', \tau_2' = \texttt{CROSS}(a_1, b_1, a_2, b_2; \tau_1, \tau_2)$$
$$\tau_1' \triangleq \tau_1[:a_1] \oplus \tau_2[a_2 : b_2] \oplus \tau_1[b_1 :]$$
$$\tau_2' \triangleq \tau_2[:a_2] \oplus \tau_1[a_1 : b_1] \oplus \tau_2[b_2 :]$$

- τ_i: input tour of vehicle i
- τ_i': updated tour of vehicle i
- $\tau_i[a:b]$: sub tour of τ_i ranging from $(a\ to\ b)$

- General representation of inter−agent local search
- CE selects the sub-tours (i.e., $\tau_1[a_2:b_2], \tau_1[a_2:b_2]$) from τ_1, τ_2 and swaps the sub-tours to generate new tours τ_1', τ_2'.
- CE seeks to find the four points (a_1, b_1, a_2, b_2) to reduce the cost of the tours.
  - $\max(C(\tau_1'), C(\tau_2')) \leq \max(C(\tau_1), C(\tau_2))$
- When the full search method is naively employed, the search cost is $O(n^4)$ where n is the number of nodes in a tour.

Solution approach: Overall solution approach



- Initial solution construction:
  - Greedy assignment heuristics to obtain the initial tours.
- Select tours:
  - Set τ_1, τ_2 as the tours of the largest and smallest among vehicles.
- Neuro Cross:
  - Utilize the cost-decrement prediction model f_θ and two-stage search method to find the cost improving tour pair with $O(n^2)$ budget.
- Intra operation:
  - For our targeting VRPs, the intra operation is equivalent to solve traveling salesman problem (TSP).

Solution approach : Neuro Cross operation

- Predicting cost decrement
  - Cost-decrement model $f_\theta(a_1, a_2;\ \tau_1, \tau_2)$ that predicts the **maximum cost decrements** from the given τ_1 and τ_1 and the starting nodes $a_1$and a_2 of their sub-tours
  - $f_\theta(a_1, a_2;\ \tau_1, \tau_2)$ supervise the Optimal decrement cost $y^*(a_1, a_2;\ \tau_1, \tau_2)$:
    - $f_\theta(a_1, a_2;\ \tau_1, \tau_2) \approx y^*(a_1, a_2;\ \tau_1, \tau_2) = max_{b_1, b_2}(C\big(CROSS(a_1, b_1, a_2, b_2;\ \tau_1, \tau_2)\big) - C(\tau_1, \tau_2))$
- Constructing search candidate set
  - To alleviate the prediction error, NCE selects the top K pairs of (a_1, a_2) that have the largest y^* out of all (a_1, a_2) choices.
- Performing reduced search
  - NCE finds the best (b_1, b_2) for each (a_1, a_2) in the search candidate set and select the best cost decreasing (a_1, b_1, a_2, b_2).
  - Unlike the full search of CE, the proposed NCE only performs the search for (b_1, b_2). This reduces the search cost from $O(n^4)\ to\ O(n^2)$

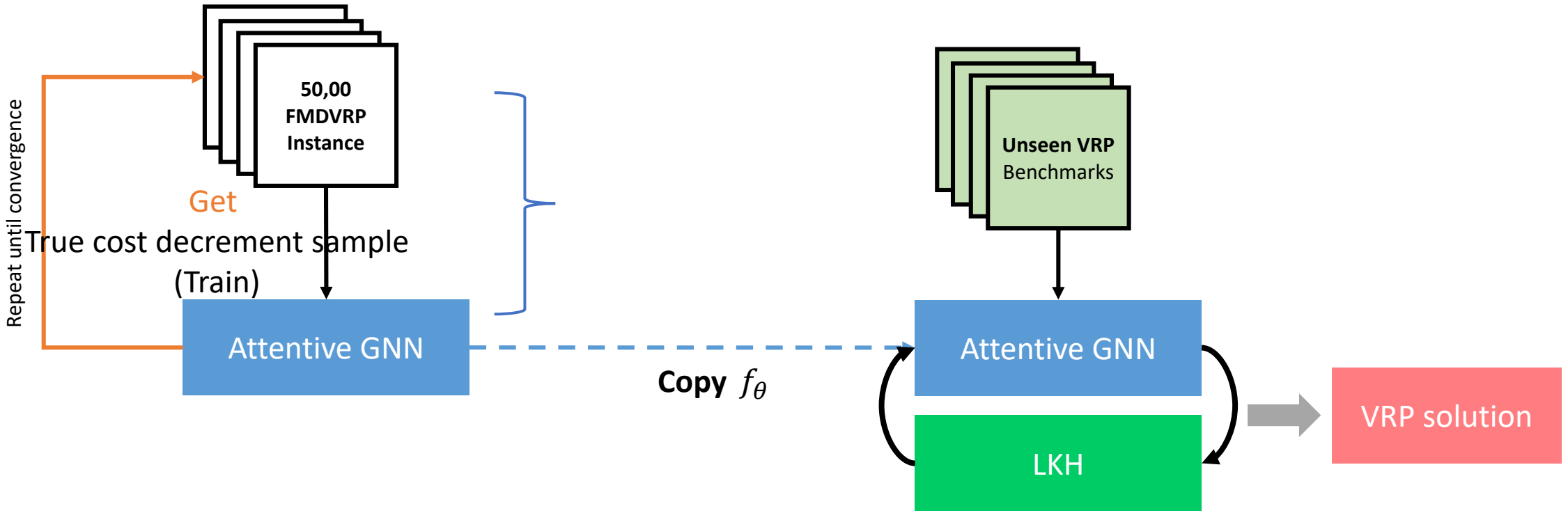
Training

<Train >

- Random (N_c, N_d), $N_v = 2$
  - 50,000 FMDVRP instance
  - 47,856,986 training labels

<Test >

- Random (N_c, N_d, N_v)
  - For each (N_c, N_d, N_v) test 100 times
- Other VRP class



Experiments: min-max FMDVRP results

Table 2: **FMDVRP results** (medium-sized instances)

N_c, N_d (↓)	$N_v(\rightarrow)$ / Method	3 Cost	3 Gap(%)	3 Time(sec.)	5 Cost	5 Gap(%)	5 Time(sec.)	7 Cost	7 Gap(%)	7 Time(sec.)
(50,6)	OR-tools	2.39	15.46	2.20	1.56	10.64	2.44	1.27	6.72	2.58
	ScheduleNet	2.61	26.09	5.21	1.86	31.91	5.44	1.57	31.93	6.14
	Greedy	3.01	46.38	0.01	2.24	58.87	0.01	1.99	67.23	0.01
	Greedy + TSP	2.75	32.85	0.03	2.12	50.35	0.02	1.91	60.50	0.02
	CE	**2.07**	0.00	21.06	1.41	0.00	9.09	1.19	0.00	5.37
	NCE	2.08	0.48	1.26	**1.40**	-0.71	1.82	**1.19**	0.00	2.23
	$N_v(\rightarrow)$ / Method	5 Cost	5 Gap(%)	5 Time(sec.)	7 Cost	7 Gap(%)	7 Time(sec.)	10 Cost	10 Gap(%)	10 Time(sec.)
(100,8)	OR-tools	2.00	14.94	30.46	1.51	12.69	32.25	1.20	10.09	34.38
	ScheduleNet	2.32	33.33	35.47	1.86	38.81	36.08	1.54	41.28	41.30
	Greedy	2.82	62.07	0.04	2.37	76.87	0.05	2.00	83.49	0.05
	Greedy + TSP	2.60	49.43	0.07	2.25	67.16	0.07	1.92	76.16	0.07
	CE	**1.74**	0.00	218.46	1.34	0.00	128.40	1.09	0.00	78.56
	NCE	1.75	0.57	6.41	**1.34**	0.00	9.54	**1.09**	0.00	13.34

- NCE has a near-zero gap compared to CE.
- Revised version of Schedulenet (2021) is added to baseline.
- NCE consistently outperforms OR-tools for both the makespan and computational time.

Experiments: min-max mTSP results

N_c (↓)	N_v (→) Method	5 Cost	5 Gap(%)	5 Time(sec.)	7 Cost	7 Gap(%)	7 Time(sec.)	10 Cost	10 Gap(%)	10 Time(sec.)
50	LKH-3	**2.00**	0.00	187.46	**1.95**	0.00	249.31	1.91	0.00	170.20
	OR-tools	2.04	2.00	3.24	1.96	0.51	3.75	1.91	0.00	3.67
	DAN	2.29	14.50	0.25†	2.11	8.21	0.26†	2.03	6.28	0.30†
	ScheduleNet	2.17	8.50	1.60	2.07	6.15	1.67	1.98	3.66	1.90
	NCE	2.02	1.00	2.25	1.96	0.51	2.44	**1.91**	0.00	3.38
	NCE-mTSP	2.02	1.00	2.48	1.96	0.51	2.50	**1.91**	0.00	3.44
	N_v (→) Method	5 Cost	5 Gap(%)	5 Time(sec.)	10 Cost	10 Gap(%)	10 Time(sec.)	15 Cost	15 Gap(%)	15 Time(sec.)
100	LKH-3	**2.20**	0.00	262.85	1.97	0.00	474.78	1.98	0.00	378.90
	OR-tools	2.41	9.55	35.47	2.03	3.05	45.40	2.03	2.53	48.86
	DAN	2.72	23.64	0.43†	2.17	10.15	0.48†	2.09	5.56	0.58†
	ScheduleNet	2.59	17.73	14.84	2.13	8.12	16.22	2.07	4.55	20.02
	NCE	2.25	2.27	16.01	1.98	0.51	12.22	**1.98**	0.00	24.08
	NCE-mTSP	2.24	1.82	16.36	**1.97**	0.00	13.00	**1.98**	0.00	23.37
	N_v (→) Method	10 Cost	10 Gap(%)	10 Time(sec.)	15 Cost	15 Gap(%)	15 Time(sec.)	20 Cost	20 Gap(%)	20 Time(sec.)
200	LKH-3	**2.04**	0.00	1224.40	2,00	0.00	1147.13	1.97	0.00	908.14
	OR-tools	2.33	14.22	675.79	2.33	16.50	604.31	2.37	20.30	649.17
	DAN	2.40	17.65	0.93†	2.20	10.00	0.98†	2.15	9.14	1.07†
	ScheduleNet	2.45	20.10	193.41	2.24	12.00	213.07	2.17	10.15	225.50
	NCE	2.06	0.98	83.82	**2.00**	0.00	72.32	**1.97**	0.00	113.74
	NCE-mTSP	2.06	0.98	84.96	**2.00**	0.00	84.28	**1.97**	0.00	118.55

- NCE-mTSP is trained using mTSP instance
- NCE is outperform than other baselines
- NCE is significantly faster than LKH3

Experiments: min-max CVRP results

Table 5: **min-max CVRP results:** $(s.n)$ indicates the best results of n sampling. The baseline results are taken from (Bogyrbayeva et al., 2021).

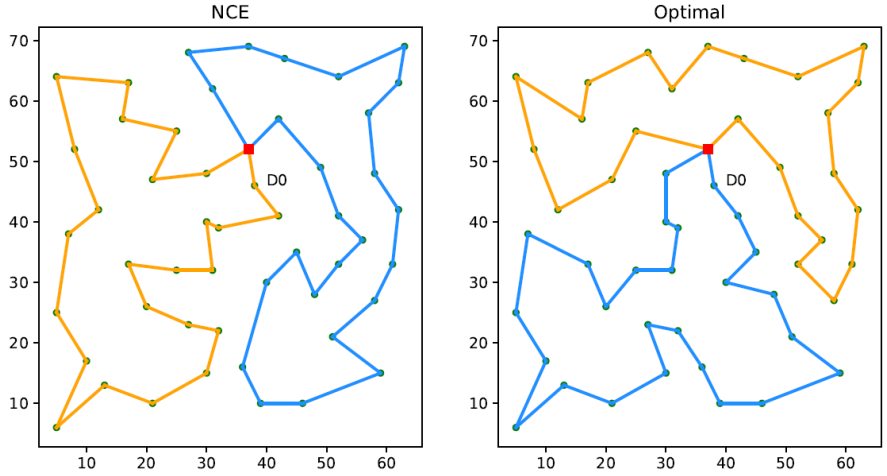
$N_c, N_v(\rightarrow)$	*20,3*			*30,3*		
Method	Cost	Gap(%)	Time(sec.)	Cost	Gap(%)	Time(sec.)
OR-tools	2.04	0.99	1.0	2.44	11.42	1.0
AM (Bogyrbayeva et al., 2021)	2.20	8.91	0.1	2.47	12.79	0.2
AM (s.1200) (Bogyrbayeva et al., 2021)	2.44	20.79	11.4	2.29	4.57	27.6
HM (Bogyrbayeva et al., 2021)	2.28	12.87	0.1	2.39	9.13	0.2
HM (s.1200) (Bogyrbayeva et al., 2021)	2.15	6.44	14.3	2.27	3.65	25.2
NCE	2.06	1.98	1.16	2.25	2.74	2.03
NCE(s.10)	**2.02**	0.00	2.31	**2.19**	0.00	5.78

- NCE with the f_θ trained on FMDVRP instances to solve CVRP
- Even though training f_θ is done without the consideration of the capacity constraints, we can easily enforce such constraints without retraining

$$(b_1, b_2) \leftarrow \underset{b_1, b_2 \in S_c}{\arg\max} \left(C(\texttt{CROSS}((a_1, b_1, a_2, b_2; \tau_1, \tau_2))) - C(\tau_1, \tau_2) \right)$$

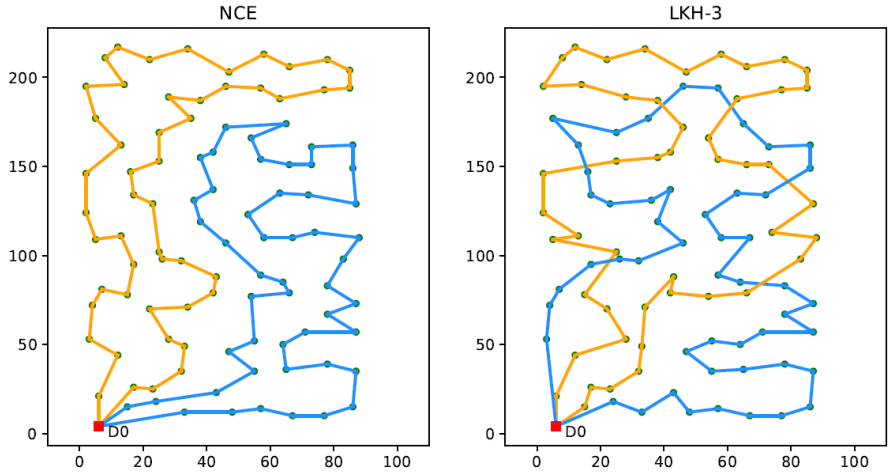
Experiments: Vehicle route examples

mTSP lib Eil51 -2 result

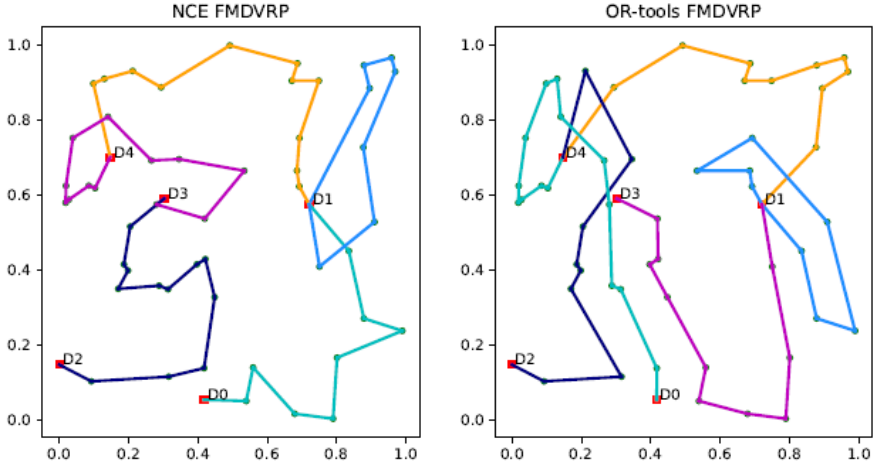


Solution value: NCE 226.1, Optimal 222.7

mTSP lib Rat99 -2 result

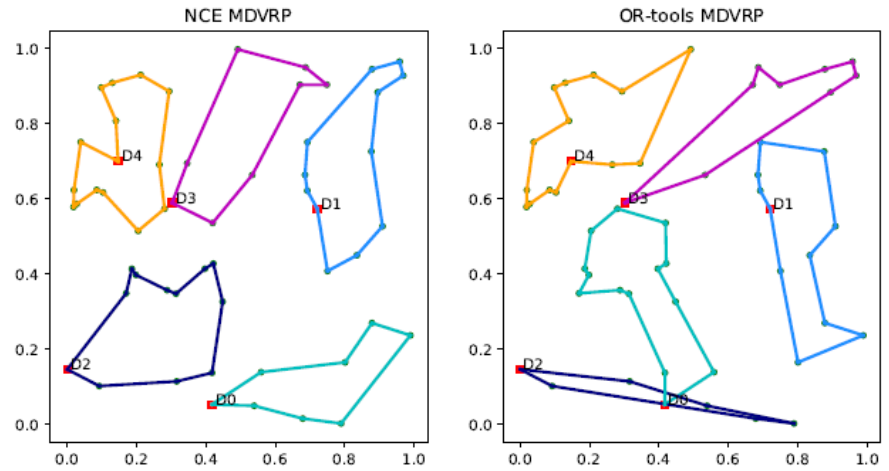


Solution value: NCE 666.0, LKH-3 728.8



FMDVRP solutions computed by NCE and OR-tools

Solution value: NCE 1.35, OR-tools 1.52



MDVRP solutions computed by NCE and OR-tools

Solution value: NCE 1.39, OR-tools 1.61

End of presentation

Thank you for your listening