Pareto-NRPA: A Novel Monte-Carlo Search Algorithm for Multi-Objective Optimization

28th European Conference on Artificial Intelligence (ECAI 2025)

27/10/2025

Noé Lallouet[1 2], Tristan Cazenave[2], Cyrille Enderli[1]

[1] Thales DMS, [2] Université Paris-Dauphine - PSL

Introduction

Multi-objective optimization (MOO) problem

$$\text{minimize}_x \quad f_1(x), f_2(x), \dots, f_p(x)$$
$$\text{subject to} \quad x \in \mathcal{X}$$

- $f_1, \dots, f_p$ are p objective functions
- $\mathcal{X}$ is the feasible set
- Pareto-dominance: a solution x dominates y ($x \prec y$) iff:
  - $f_i(x) \leq f_i(y), \forall i = 1, \dots, p$
  - $\exists j \in 1, \dots, p : f_j(x) < f_j(y)$
- A solution x is said Pareto-optimal iff:
  - $\neg \exists y \in \mathcal{X} : y \prec x$

Problem scope

- Discrete search spaces:
  - Finite set of states and actions
- Possible constraints
  - Certain actions may be unfeasible in a particular state.

Contributions

- We propose Pareto-NRPA, a multi-objective optimization algorithm generalizing NRPA (Nested Rollout Policy Adaptation)[1]
- We introduce a novel benchmark for constrained discrete multi-objective optimization: MO-TSPTW
- We benchmark Pareto-NRPA on two sets of problems:
  - All instances of MO-TSPTW
  - Two neural architecture search (NAS) datasets.

[1] Christopher Rosin, "Nested Rollout Policy Adaptation for Monte Carlo Tree Search," 2011, 649–54, https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-115.

NRPA algorithm

NRPA(level, π)

```
if level=0 then
    return playout(root, π)
else
    bestScore ← ∞
    for N iterations do
        (result, new) ← NRPA(level−1,π)
        if result ≤ bestScore then
            bestScore ← result
            seq ← new
        end if
        π ← Adapt(π, seq)
    end for
    return (bestScore, seq)
end if
```

- NRPA runs nested searches in a recursive fashion, starting from an initial level L.
- At each level ℓ, NRPA runs N iterations of level $\ell - 1$ searches
- At level 0, the search is a playout guided by a policy π
- After each search, the policy is updated with respect to the best sequence found so far.
- The adapt algorithm updates the policy weights by performing gradient ascent towards the actions of the best sequence found during the search

NRPA algorithm

NRPA(level, π)

if level=0 **then**
  return playout(*root*, π)
else
  bestScore $\leftarrow \infty$
  for N iterations **do**
    (*result*, *new*) $\leftarrow$ NRPA(*level*−1, π)
    if *result* $\leq$ *bestScore* **then**
      bestScore $\leftarrow$ *result*
      seq $\leftarrow$ *new*
    end if
    $\pi \leftarrow$ Adapt(π, *seq*)
  end for
  return (*bestScore*, *seq*)
end if

Adapt(π, sequence)

$\pi' \leftarrow \pi$
state $\leftarrow$ *root*
for *move* $\in$ *sequence* **do**
  $\pi'[\text{code}(move)] \leftarrow \pi'[\text{code}(move)] + \alpha$
  $z \leftarrow 0$
  for $m \in$ possible moves for *state* **do**
    $z \leftarrow z + \exp(\pi[\text{code}(m)])$
  end for
  for $m \in$ possible moves for *state* **do**
    $\pi'[\text{code}(m)] \leftarrow \pi'[\text{code}(m)] - \alpha * \frac{\exp(\pi[\text{code}(m)])}{z}$
  end for
  state $\leftarrow$ play(*state*, *move*)
end for
$\pi \leftarrow \pi'$

Generalizing to multiple objectives

- NRPA has enjoyed success on various problems, such as Morpion Solitaire, RNA folding, routing problems…

NRPA for multi-objective optimization

Can we generalize the NRPA algorithm to approximate a Pareto front of solutions rather than a single best solution?

- Goal of Pareto-NRPA: identifying a set of optimal sequences S^* from the search space S:

$$S^* = \{s \in S \mid \neg \exists u \in S : u \neq s, u \prec s\}$$

Generalizing to multiple objectives

- **Changing the number of policies**
  - Instead of a single policy π, we use a set of policies Π
  - $\Pi = \{\pi_1, \pi_2, ..., \pi_k\}$
- By using several policies instead of one, the algorithm is able to optimize different regions of the search space, with each policy focusing on its own region.
- During a playout (at level 0):
  - The algorithm randomly chooses a policy $\pi_k \in \Pi$
  - Policy π_k is used to sample a solution s
  - We memorize $s.\text{policy} = \pi_k$

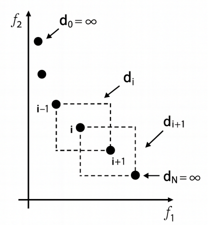
Generalizing to multiple objectives

- **Maintaining a Pareto front**
  - After a level $\ell - 1$ search, instead of memorizing the best score and its associated solution, Pareto-NRPA memorizes a non-dominated set.
  - We use Non-Dominated Sorting (NDS)[2] to efficiently identify the solutions corresponding to optimal trade-offs.

[2] K. Deb et al., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation* 6, no. 2 (2002): 182–97, https://doi.org/10.1109/4235.996017.

Generalizing to multiple objectives

- **Adapting the policy**
  - The weights of the policy are updated for each solution belonging to the non-dominated set S^*
  - We perform a weighted gradient ascent step conditioned by:
    - α, the learning rate of Pareto-NRPA
    - $D(s)$: the crowding distance of solution s with respect to the solutions of S^*
  - The crowding distance approximates the perimeter of the hyper-rectangle defined by the nearest neighbors of a point in objective space, giving a measure related to the relative isolation of a solution.



Pareto-NRPA

Pareto-NRPA(level, Π)

if *level* = 0 **then**
  choose $\pi_k : p(\pi_k) \sim U(0, |\Pi|)$
  return Playout(*root*, π_k)
else
  $S^* \leftarrow \emptyset$
  for N iterations **do**
    set ← Pareto-NRPA(*level*−1, Π)
    $S^* \leftarrow S^* \cup$ *set*
    $S^* \leftarrow \{s \in S^* \mid \neg\exists u \in S^* : u \neq s, u \prec s\}$
    Π ← Pareto-Adapt(Π, S^*)
  end for
  return S^*
end if

Pareto-Adapt(Π, S^*)

D ← CrowdingDistance(S^*)
for $s \in S^*$ **do**:
  $\pi \leftarrow s$.policy
  $\pi' \leftarrow \pi$
  state ← *root*
  for *move* ∈ *sequence* **do**
    $\pi'[\text{code}(move)] \leftarrow \pi'[\text{code}(move)] + (\alpha * D[s])$
    $z \leftarrow 0$
    for $m \in$ possible moves for *state* **do**
      $z \leftarrow z + \exp(\pi[\text{code}(m)])$
    end for
    for $m \in$ possible moves for *state* **do**
      $\pi'[\text{code}(m)] \leftarrow \pi'[\text{code}(m)] - (\alpha * \frac{\exp(\pi[\text{code}(m)])}{z} * D[s])$
    end for
    state ← play(*state*, *move*)
  end for
  $\pi \leftarrow \pi'$
  return Π

Experimental results

- Pareto-NRPA is benchmarked on two problem classes
  - A novel benchmark dataset for multi-objective constrained traveling salesman problem: MO-TSPTW
  - Neural Architecture Search (NAS) benchmarks:
    - NAS-Bench-201[3]
    - NAS-Bench-101[4]

---

[3] Xuanyi Dong and Yi Yang, *NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search*, technical report no. arXiv:2001.00326 (2020), http://arxiv.org/abs/2001.00326.

[4] Chris Ying et al., *NAS-Bench-101: Towards Reproducible Neural Architecture Search*, technical report no. arXiv:1902.09635 (2019), http://arxiv.org/abs/1902.09635.

Experimental results

- Pareto-NRPA is compared to:
  - Leading multi-objective evolutionary algorithms:
    - NSGA-II[5]
    - SMS-EMOA[6]
    - MOEA/D[7]
  - Pareto Local Search (PLS)[8]
  - Pareto-MCTS[9]

---

[5] Deb et al., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II".

[6] Nicola Hochstrate, Boris Naujoks, and Michael Emmerich, "SMS-EMOA: Multiobjective Selection Based on Dominated Hypervolume," *European Journal of Operational Research* 181 (February 2007): 1653–69, https://doi.org/10.1016/j.ejor.2006.08.008.

[7] Qingfu Zhang and Hui Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Transactions on Evolutionary Computation* 11, no. 6 (December 2007): 712–31, https://doi.org/10.1109/TEVC.2007.892759.

[8] Luis Paquete, Marco Chiarandini, and Thomas Stützle, "Pareto Local Optimum Sets in the Biobjective Traveling Salesman Problem: An Experimental Study," in *Metaheuristics for Multiobjective Optimisation*, *Metaheuristics for Multiobjective Optimisation*, ed. Xavier Gandibleux et al. (Berlin, Heidelberg: Springer, 2004), 177–99, https://doi.org/10.1007/978-3-642-17144-4_7.

[9] Weizhe Chen and Lantao Liu, "Pareto Monte Carlo Tree Search for Multi-Objective Informative Planning," in "Robotics: Science and Systems XV," special issue, *Robotics: Science and Systems XV*, Robotics: Science, Systems Foundation, June 2019, https://doi.org/10.15607/RSS.2019.XV.072.

Evaluation metrics

Hypervolume: $HV(Y^*, r) = \lambda_m\left(\bigcup_{y\in Y^*}[y, r]\right)$

- Represents the volume of the space dominated by Y^* and bounded by a reference point r, where λ_m is the Lebesgue measure in m dimensions.

Overall Spread (OS): $OS(Y^*) = \prod_{i=1}^{p} \frac{\left|\max\limits_{y\in Y^*} y_i - \min\limits_{y\in Y^*} y_i\right|}{\left|\tilde{y}_i^l - \tilde{y}_i^M\right|}$

- Represents the extent that the Pareto front covers in the objective space

Spacing: $\sqrt{\frac{1}{\mid Y^* - 1 \mid} \sum_{j=1}^{\mid Y^* \mid} \left(\bar{d} - d^1(y^j, Y^* \setminus \{y^j\})^2\right)}$

- Related to the variation of the Manhattan distance between elements of the Pareto front.

MO-TSPTW

- Multi-objective extension of the Solomon-Potvin-Bengio TSPTW dataset[10]
- Objective: find the shortest tour of n cities
  - Each city has to be visited within a specific time window to be valid
- MO-TSPTW: we generate a secondary cost matrix that acts as the second objective to optimize.

$$\text{minimize} \quad \begin{cases} f_1(s) = \text{cost}_1(s) + 10^6 \times \Omega(s) \\ f_2(s) = \text{cost}_2(s) + 10^6 \times \Omega(s) \end{cases}$$

[10] Jean-Yves Potvin and Samy Bengio, "The Vehicle Routing Problem with Time Windows Part II: Genetic Search," *INFORMS Journal on Computing* 8, no. 2 (May 1996): 165–72, https://doi.org/10.1287/ijoc.8.2.165.

MO-TSPTW

- MO-TSPTW is available online:
  - https://github.com/pareto-nrpa/mo-tsptw

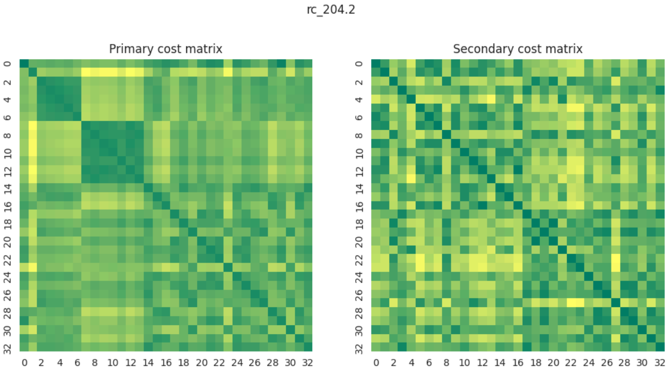


Figure 2: The two independent cost matrices for instance `rc_204.2` of MO-TSPTW

MO-TSPTW

- Results on three selected instances of MO-TSPTW:
- Allocated budget: 100 000 objective function evaluations.

Algorithm	HV	CV
NSGA-II	**0.97**	**0.00**
SMS-EMOA	0.96	**0.00**
Pareto-MCTS	0.61	**0.00**
PLS	0.91	**0.00**
MOEA/D	0.93	**0.00**
Pareto-NRPA	0.94	**0.00**

Algorithm	HV	CV
NSGA-II	0.00	7.33
SMS-EMOA	0.00	5.97
Pareto-MCTS	0.00	9.63
PLS	0.00	11.80
MOEA/D	0.00	5.53
Pareto-NRPA	**0.91**	**0.00**

Algorithm	HV	CV
NSGA-II	0.12	1.90
SMS-EMOA	0.00	2.17
Pareto-MCTS	0.00	20.47
PLS	0.00	5.67
MOEA/D	0.01	2.23
Pareto-NRPA	**0.26**	**0.00**

● rc_204.3: largest average time window

● rc_201.3: narrowest average time window

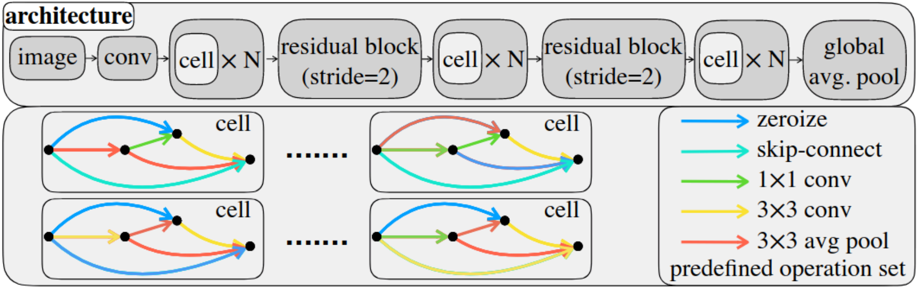
● rc_204.1: largest number of cities

MO-TSPTW

- Pareto-NRPA obtains the highest hypervolume values on 22 of the 31 instances of MO-TSPTW, and particularly on 20 of the 21 **hardest instances**.
- The policy adaptation mechanism implemented in Pareto-NRPA allows the algorithm to learn sequences of moves which leads to faster convergence.
- Pareto-NRPA is an efficient algorithm for constraint handling in sequential discrete multi-objective optimization problems, strongly outperforming MOEAs on problems with constraints that are hard to satisfy.

Neural Architecture Search

- Search space: sequences of operations that create a *network cell*
- A neural network consists of the same *cell* stacked according to a predefined skeleton



Neural Architecture Search

NAS-Bench-201:

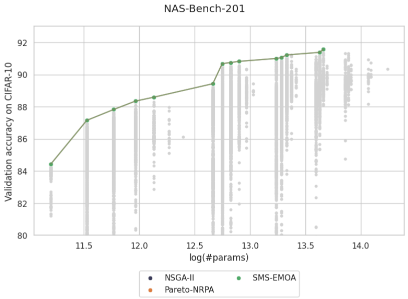
- $\sim$ 16k architectures

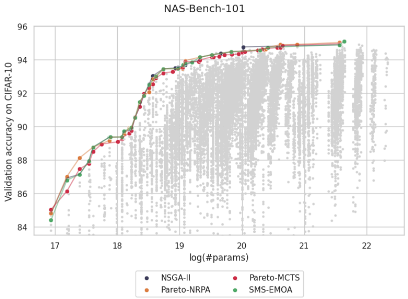
Algorithm	Hypervolume	Overall Spread
NSGA-II	$\mathbf{0.99 \pm 0.00}$	$\mathbf{0.72 \pm 0.02}$
SMS-EMOA	$\mathbf{0.99 \pm 0.00}$	0.70 ± 0.03
Pareto-MCTS	0.96 ± 0.00	0.68 ± 0.05
Pareto-NRPA	0.98 ± 0.00	0.70 ± 0.03

NAS-Bench-101:

- $\sim$ 423k architectures

Algorithm	Hypervolume	Overall Spread
NSGA-II	0.98 ± 0.00	0.64 ± 0.04
SMS-EMOA	0.97 ± 0.00	0.62 ± 0.03
Pareto-MCTS	0.97 ± 0.00	0.56 ± 0.03
Pareto-NRPA	$\mathbf{0.99 \pm 0.00}$	$\mathbf{0.72 \pm 0.04}$





Conclusion

- We have proposed Pareto-NRPA, a multi-objective optimization algorithm based on Monte-Carlo search
- Pareto-NRPA is shown to perform strongly on constrained sequential discrete multi-objective optimization problems
- Performance is validated on a NAS benchmark dataset and on a novel MO-TSPTW dataset.

Thanks for your attention!

link to the paper

References

[1] C. Rosin, “Nested Rollout Policy Adaptation for Monte Carlo Tree Search,” 2011, pp. 649–654. doi: 10.5591/978-1-57735-516-8/IJCAI11-115.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002, doi: 10.1109/4235.996017.

[3] X. Dong and Y. Yang, “NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search,” technical report arXiv:2001.00326, Jan. 2020. Accessed: Nov. 30, 2023. [Online]. Available: http://arxiv.org/abs/2001.00326

[4] C. Ying, A. Klein, E. Real, E. Christiansen, K. Murphy, and F. Hutter, “NAS-Bench-101: Towards Reproducible Neural Architecture Search,” technical report arXiv:1902.09635, May 2019. Accessed: June 26, 2024. [Online]. Available: http://arxiv.org/abs/1902.09635

[5] N. Hochstrate, B. Naujoks, and M. Emmerich, “SMS-EMOA: Multiobjective selection based on dominated hypervolume,” *European Journal of Operational Research*, vol. 181, pp. 1653–1669, Feb. 2007, doi: 10.1016/j.ejor.2006.08.008.

[6] Q. Zhang and H. Li, “MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition,” *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, Dec. 2007, doi: 10.1109/TEVC.2007.892759.

[7] L. Paquete, M. Chiarandini, and T. Stützle, “Pareto Local Optimum Sets in the Biobjective Traveling Salesman Problem: An Experimental Study,” in *Metaheuristics for Multiobjective Optimisation*, X. Gandibleux, M. Sevaux, K. Sörensen, and V. T’kindt, Eds., Berlin, Heidelberg: Springer, 2004, pp. 177–199. doi: 10.1007/978-3-642-17144-4_7.

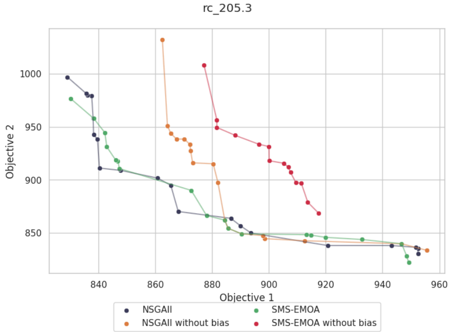
[8] W. Chen and L. Liu, “Pareto Monte Carlo Tree Search for Multi-Objective Informative Planning,” in *Robotics: Science and Systems XV*, Robotics: Science, Systems Foundation, June 2019. doi: 10.15607/RSS.2019.XV.072.

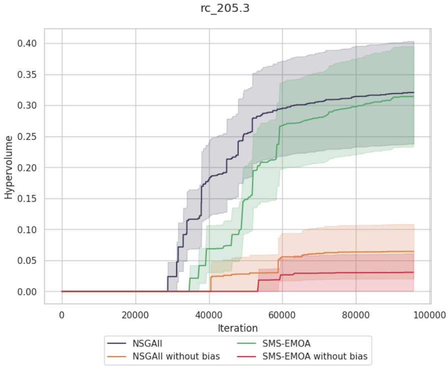
[9] J.-Y. Potvin and S. Bengio, “The Vehicle Routing Problem with Time Windows Part II: Genetic Search,” *INFORMS Journal on Computing*, vol. 8, no. 2, pp. 165–172, May 1996, doi: 10.1287/ijoc.8.2.165.

Appendix

Impact of bias

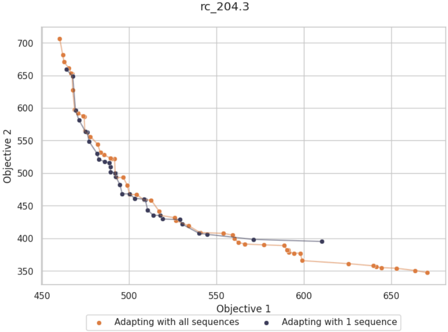
- Bias term: $-10 \times d_{ij}$, with d_{ij} the distance between two cities
- Using a bias improves the EMOAs.

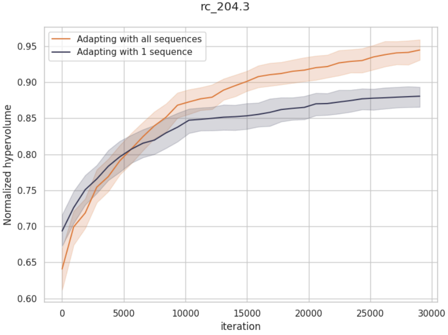




Adapting the policy with one or all sequences

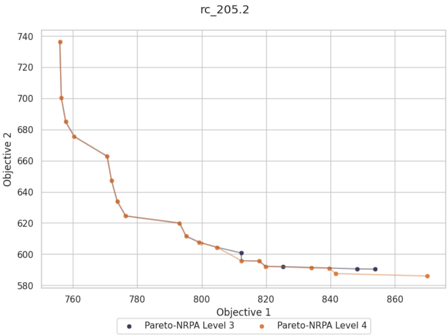
- We investigate the effect of updating the policy with:
  - All sequences, weighted by their crowding distance
  - The sequence that maximizes the crowding distance

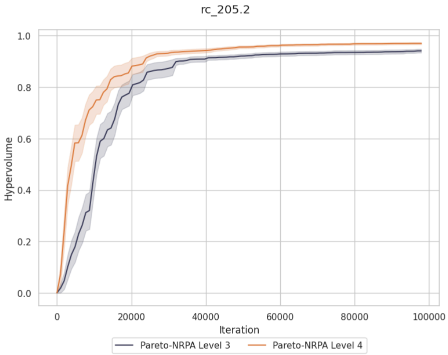




Impact of NRPA level

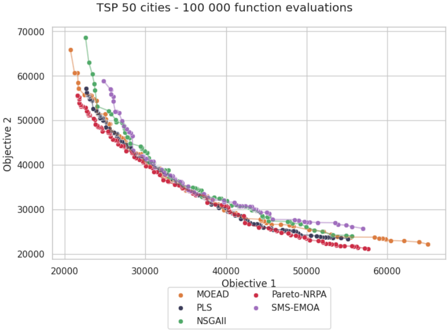
- With the same number of function evaluations, we evaluate Pareto-NRPA with levels 3 and 4.

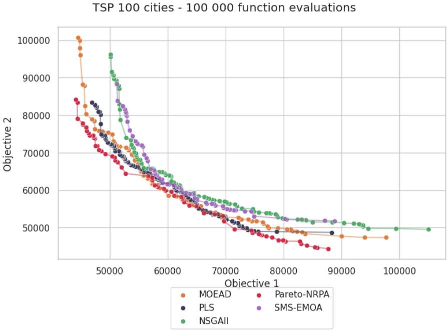




Benchmark on MO-TSP

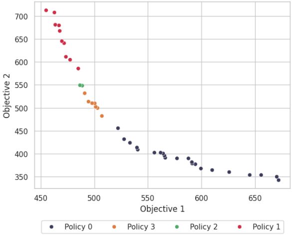
- Multi-objective Traveling Salesman Problem
- Pareto-NRPA still performs extremely well

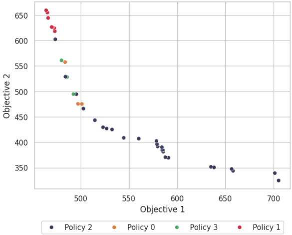




Policy convergence towards regions of the objective space

- Two different runs on rc_204.3
- One run has converged to policies focusing on different regions of the search space, leading to varied solutions, while the other has a less distinct separation of policies in the solution space.
- Improving Pareto-NRPA's distribution of policies over different regions of the search space remains an axis for future work.





Results on MO-TSPTW

Instance	Cities	NSGA-II	SMS-EMOA	Pareto-UCT	Pareto-NRPA
rc_206.1	4	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
rc_207.4	6	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
rc_203.4	15	**1.00 ± 0.00**	**1.00 ± 0.00**	0.65 ± 0.01	0.95 ± 0.01
rc_202.2	14	**1.00 ± 0.00**	**1.00 ± 0.00**	0.84 ± 0.01	0.98 ± 0.00
rc_204.3	24	**0.97 ± 0.01**	0.96 ± 0.01	0.61 ± 0.02	0.94 ± 0.01
rc_203.1	19	0.91 ± 0.02	0.87 ± 0.03	0.44 ± 0.04	**0.97 ± 0.01**
rc_204.2	33	0.80 ± 0.03	0.79 ± 0.04	0.00 ± 0.00	**0.82 ± 0.02**
rc_208.2	29	**0.94 ± 0.01**	0.93 ± 0.01	0.00 ± 0.00	0.86 ± 0.02
rc_204.4	14	**1.00 ± 0.00**	**1.00 ± 0.00**	0.60 ± 0.04	0.99 ± 0.01
rc_205.1	14	**1.00 ± 0.00**	**1.00 ± 0.00**	0.63 ± 0.04	**1.00 ± 0.01**
rc_204.1	46	0.12 ± 0.11	0.00 ± 0.00	0.00 ± 0.00	**0.29 ± 0.05**
rc_203.2	33	0.11 ± 0.08	0.27 ± 0.12	0.00 ± 0.00	**0.86 ± 0.02**
rc_203.3	37	0.08 ± 0.07	0.04 ± 0.05	0.00 ± 0.00	**0.90 ± 0.02**
rc_208.3	36	0.92 ± 0.06	**0.93 ± 0.01**	0.00 ± 0.00	0.81 ± 0.02
rc_202.4	28	0.01 ± 0.02	0.07 ± 0.06	0.00 ± 0.00	**0.77 ± 0.08**
rc_208.1	38	0.06 ± 0.07	0.06 ± 0.07	0.00 ± 0.00	**0.46 ± 0.06**
rc_207.3	33	0.24 ± 0.12	0.28 ± 0.14	0.00 ± 0.00	**0.92 ± 0.02**
rc_207.2	31	0.00 ± 0.00	0.08 ± 0.09	0.00 ± 0.00	**0.33 ± 0.11**
rc_206.3	25	0.56 ± 0.17	0.63 ± 0.16	0.01 ± 0.01	**0.94 ± 0.02**
rc_207.1	34	0.42 ± 0.15	0.44 ± 0.14	0.00 ± 0.00	**0.79 ± 0.03**
rc_202.1	33	0.00 ± 0.00	0.08 ± 0.09	0.00 ± 0.00	**0.91 ± 0.02**
rc_205.3	35	0.27 ± 0.14	0.25 ± 0.13	0.00 ± 0.00	**0.77 ± 0.05**
rc_201.1	20	0.77 ± 0.15	0.73 ± 0.16	0.61 ± 0.05	**0.97 ± 0.00**
rc_205.2	27	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.92 ± 0.05**
rc_205.4	28	0.25 ± 0.13	0.28 ± 0.14	0.00 ± 0.00	**0.80 ± 0.03**
rc_202.3	29	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.94 ± 0.01**
rc_206.2	37	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.76 ± 0.07**
rc_206.4	38	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.51 ± 0.08**
rc_201.2	26	0.06 ± 0.08	0.25 ± 0.15	0.00 ± 0.00	**0.84 ± 0.03**
rc_201.4	26	0.00 ± 0.00	0.08 ± 0.09	0.00 ± 0.00	**0.33 ± 0.11**
rc_201.3	32	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.93 ± 0.02**