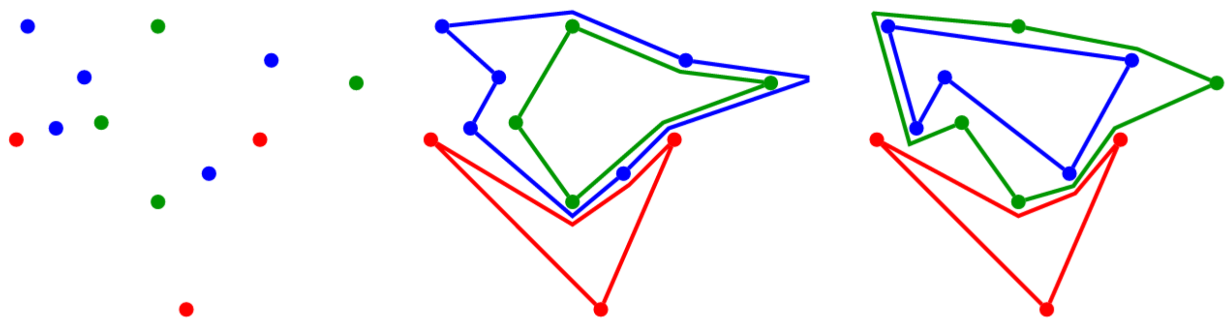
A ($\frac{5}{3}+\epsilon$) - Approximation for Tricolored Non-crossing Euclidean TSP

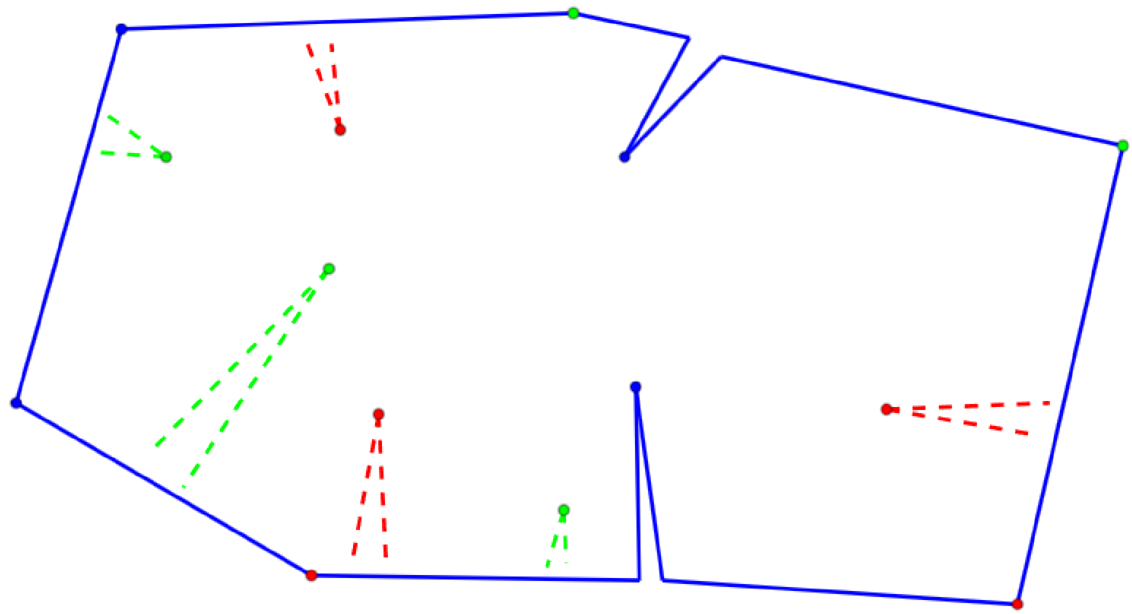
ESA2024

Setting

- k-ETSP: Given k = 3 sets of points in the Euclidea plane and are looking for disjoint tours, each covering one of the sets.
  PTAS for k = 1 (1998) and k = 2 (2023) based on "patching"
- The objective of k-ETSP is to minimize the total length of the tours, i.e., to minimize $l(\Pi) = \sum_{c \in C} l(\pi_c)$ strictly $OPT^* := inf\{l(\Pi)$: Π is a solution$\}$



obviously such tours always exists and OPT should consist of straight line.



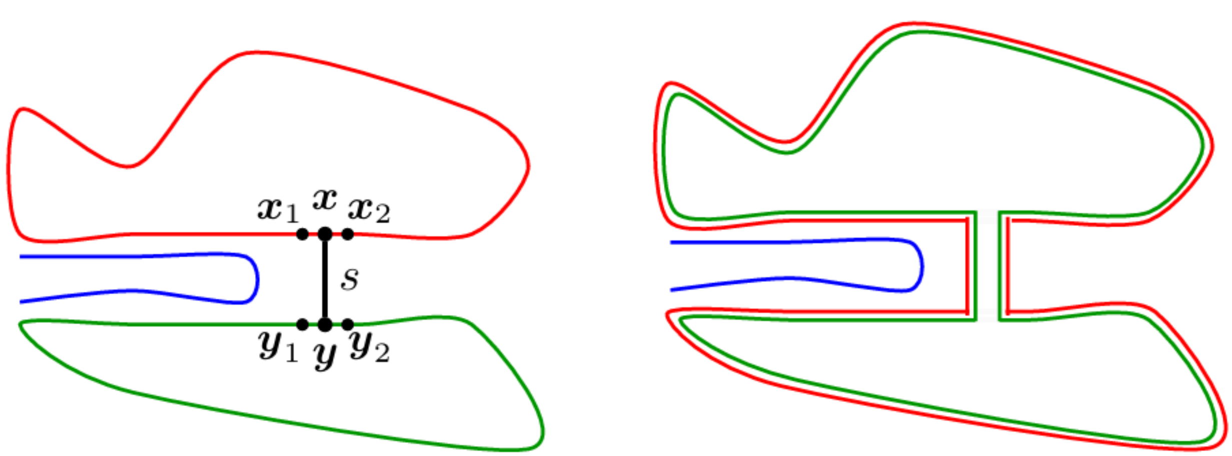
Result

For every $\epsilon > 0$,there is an algorithm that computes a $(\frac{5}{3} + \epsilon)$-approximation for 3-ETSP in time $(\frac{n}{\epsilon})^{O(\frac{1}{\epsilon})}$

Main Technoloy

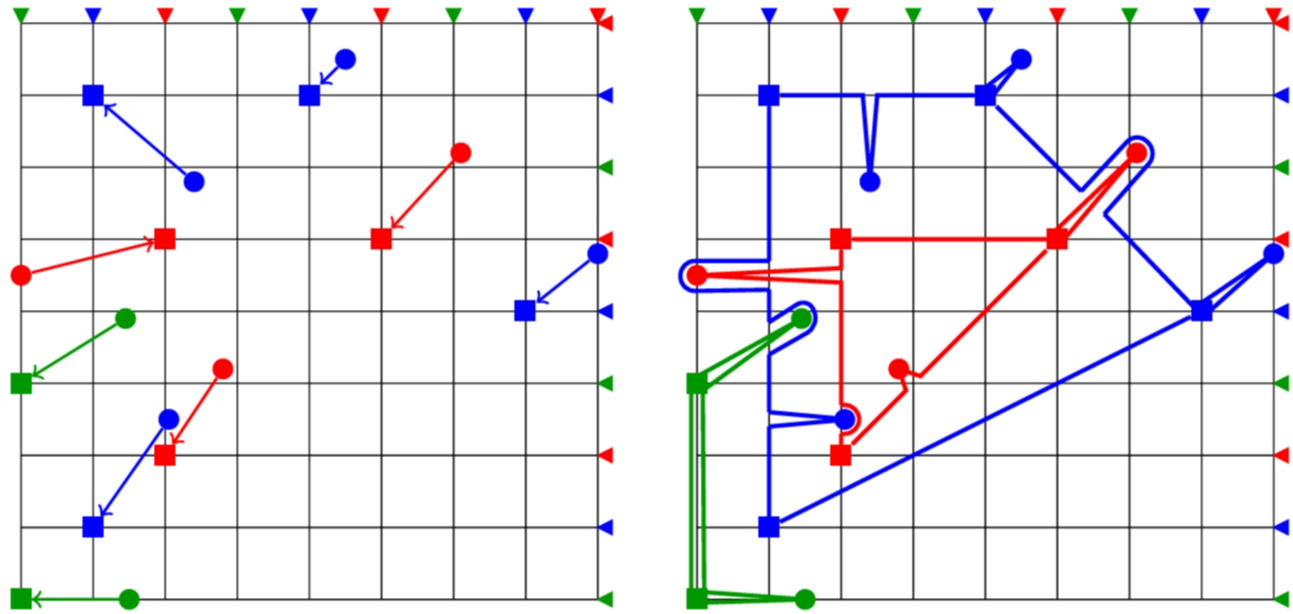
Idea: $OPT \xrightarrow{cost} OPT^* \xleftarrow{polynomial} ALG$

Ratio of $\frac{5}{3}$:



Dissection

$R^2 \rightarrow \{0, 1, 2, ..., L\}^2, L = 2^l$ with extra costs of O($\epsilon' OPT$)

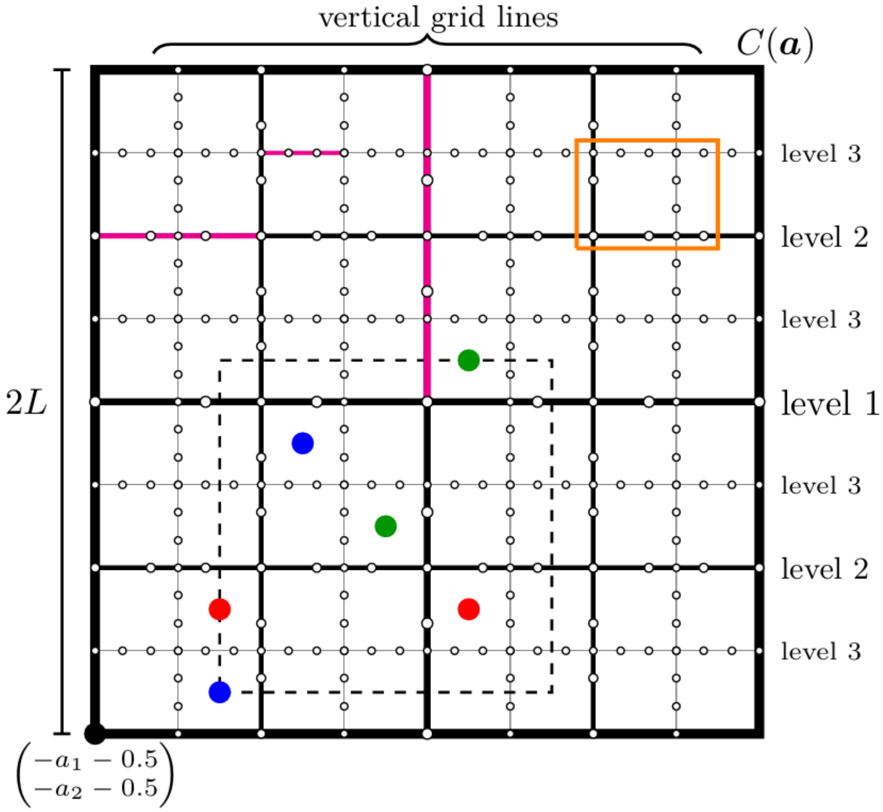


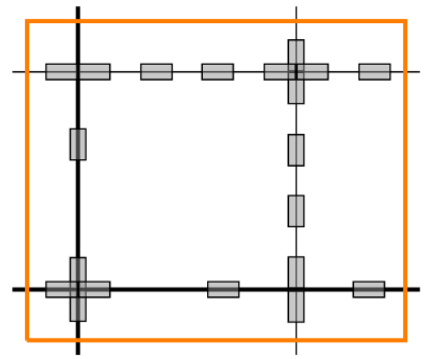
Structure and estimated expectations

Fix an instance of k-ETSP with $T \subseteq \{0, ..., L\}^2$ where L is a power of two. We pick a shift vector a = (a1,a2) $\in \{0, ..., L-1\}^2$ and consider the square

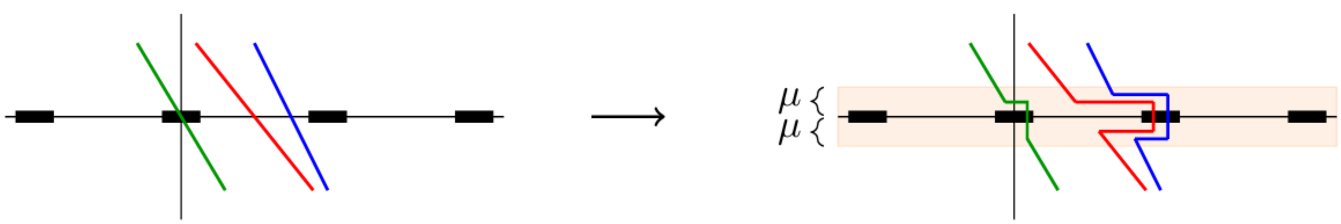
$$C(a) := [-a_1 - 0.5 \; , 2L - a_1 - 0.5] \times [-a_2 - 0.5, \; 2L - a_2 - 0.5]$$

i.e., $C(a)$ is the square $[0, ..., 2L]^2$ shifted by $-a - (0.5, 0.5)$
then set $portals$ on $boundary$





Adjustments to OPT：

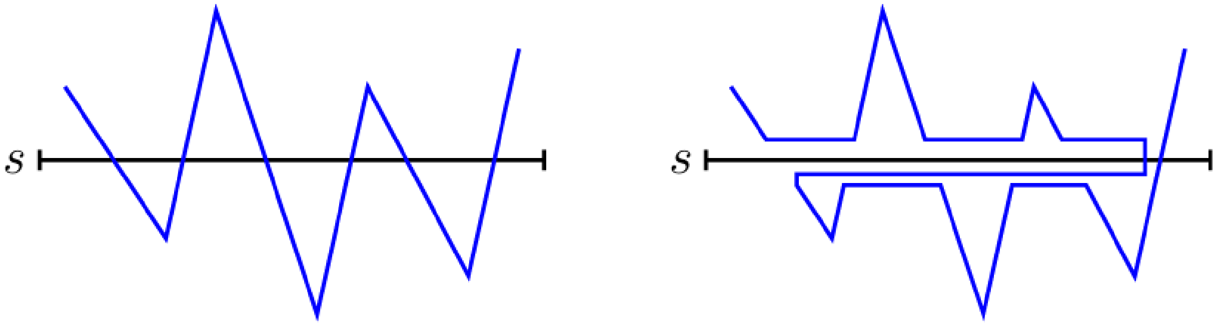


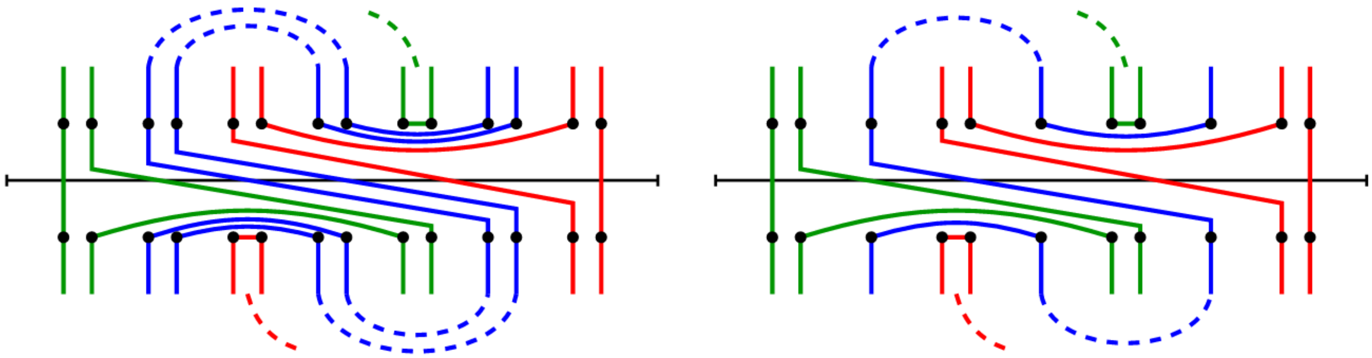
$$\mathbb{E}_{\boldsymbol{a}}\left[l(\mathcal{S}') - l(\mathcal{S})\right] \leq 7\sqrt{2} \cdot \frac{l(S)}{r},\ where\ l(\mathcal{S}) := \sum_{s \in \mathcal{S}} l(s).$$

Patching

Restrict the route of the curve at $portals$

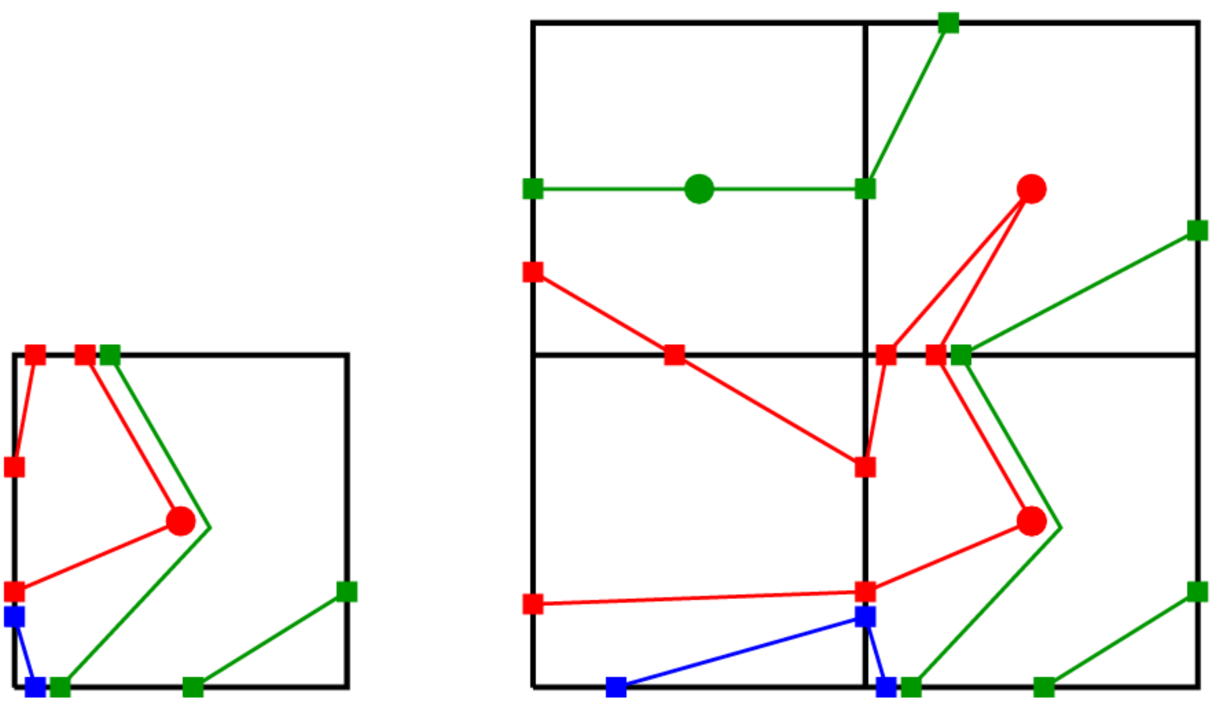
Specifically, limit the number(18 times) of intersections with the $portals$





Algorithm

Search paths:



Algorithm 1 Perturbation

input: $\varepsilon > 0$, $T_{\mathrm{R}}, T_{\mathrm{G}}, T_{\mathrm{B}} \subseteq \mathbb{R}^2$ with $\left|\bigcup_{c\in\{\mathrm{R,G,B}\}} T_c\right| =: n$

$(S, f) \leftarrow \operatorname{argmin}\{f' : S'$ is a square of side length f such that $\bigcup_{c\in\{\mathrm{R,G,B}\}} T_c \subseteq S\}$

$L \leftarrow \min\{2^i : 2^i \geq \lceil 3 \cdot 117 \cdot n/\varepsilon \rceil, i \in \mathbb{N}\}$

let $v_0, \dots, v_L$ and $h_0, \dots, h_L$ denote $L+1$ vertical and $L+1$ horizontal equispaced straight line segments of length f in S with $h_0, v_0, h_L, v_L \subseteq \partial S'$

$T'_{\mathrm{R}}, T'_{\mathrm{G}}, T'_{\mathrm{B}} \leftarrow \emptyset$

forall $c \in \{\mathrm{R, G, B}\}$ *and* $\boldsymbol{t} \in T_c$ **do**

$$k \leftarrow \begin{cases} 0, & \text{if } c = R \\ 1, & \text{if } c = G \\ 2, & \text{if } c = B \end{cases}$$

$\boldsymbol{t'} \leftarrow \operatorname{argmin}\{\|\boldsymbol{t''} - \boldsymbol{t}\| : \boldsymbol{t''} = v_i \cap h_j$ with $i \bmod 3 = j \bmod 3 = k\}$

add $\boldsymbol{t'}$ to T'_c

identify the points $\{v_i \cap h_j : i, j \in \{0, \dots, L\}\}$ with $\{0, \dots, L\}^2$

return $L, T'_{\mathrm{R}}, T'_{\mathrm{G}}, T'_{\mathrm{B}}$

Algorithm 2 Back-Perturbation

input: $T_{\mathrm{R}}, T_{\mathrm{G}}, T_{\mathrm{B}} \subseteq [0, L]^2$, $(1+\varepsilon')$-approximate solution $\pi'_{\mathrm{R}}, \pi'_{\mathrm{G}}, \pi'_{\mathrm{B}}$ to an instance $T'_{\mathrm{R}}, T'_{\mathrm{G}}, T'_{\mathrm{B}} \subseteq \{0, \dots, L\}^2$

choose $\delta > 0$ small enough

forall $c \in \{\mathrm{R, G, B}\}$ *and* $\boldsymbol{t} \in T_c$ **do**

$s \leftarrow \operatorname{argmin}\{l(s') : s'$ is a straight line segment between $\boldsymbol{t}$ and $\pi'_c\}$

if $s \cap \left(\left(\bigcup_{c\in\{\mathrm{R,G,B}\}} (T_c \cup T'_c)\right) \setminus \{\boldsymbol{t}\}\right) \neq \emptyset$ **then**

replace s by another segment s' connecting $\boldsymbol{t}$ and π_c such that $s' \cap \left(\left(\bigcup_{c\in\{\mathrm{R,G,B}\}} (T_c \cup T'_c)\right) \setminus \{\boldsymbol{t}\}\right) = \emptyset$ and $l(s') \leq 2l(s)$

apply patching (Lemma 13) to $\pi_{c'}, \pi_{c''}$ along s where $c', c'' \neq c$

redirect the remaining crossings around s

choose $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \pi'_c$ close enough at distance at most δ to $s \cap \pi_c$ such that $\pi_c[\boldsymbol{x}_1, \boldsymbol{x}_2] \cap T = \emptyset$

replace π'_c by $(\pi_c \setminus \pi_c[\boldsymbol{x}_1, \boldsymbol{x}_2]) \cup \overline{\boldsymbol{x_1 t}} \cup \overline{\boldsymbol{x_2 t}}$

return $\pi'_{\mathrm{R}}, \pi'_{\mathrm{G}}, \pi'_{\mathrm{B}}$

Algorithm 3 $\left(\frac{5}{3}+\varepsilon\right)$-approximation for 3-ETSP

parameters: large enough constant M
input : $\varepsilon > 0$, disjoint terminal sets $T_{\mathrm{R}}, T_{\mathrm{G}}, T_{\mathrm{B}} \subseteq \{0, \dots, L\}^2$

$(L, T'_{\mathrm{R}}, T'_{\mathrm{G}}, T'_{\mathrm{B}}) \leftarrow \textsc{Perturbation}(\varepsilon/M, T_{\mathrm{R}}, T_{\mathrm{G}}, T_{\mathrm{B}})$
choose $\delta, \mu > 0$ small enough
$\mathrm{Sol} \leftarrow \emptyset$
forall $\boldsymbol{a} \in \{0, \dots, L-1\}^2$ **do**
  compute a $(1+\mu)$-approximate solution $\Pi = (\pi_{\mathrm{R}}, \pi_{\mathrm{G}}, \pi_{\mathrm{B}})$ for 3-ETSP′ with terminals $T'_{\mathrm{R}}, T'_{\mathrm{G}}, T'_{\mathrm{B}}$ that is $(\lceil M(15\sqrt{2}+4)/\varepsilon \rceil, 18, \delta)$-portal-respecting in $D(\boldsymbol{a})$
  $\mathrm{Sol} \leftarrow \mathrm{Sol} \cup \{\Pi\}$
  forall $\{c, c', c''\} = \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$ **do**
    compute a $(1+\mu)$-approximate solution (π_1, π_2) for 2-ETSP′ with terminals $T_{c^*} := T_c \cup T_{c'}, T_{c''}$ and weights $w_{c^*} = 2, w_{c''} = 1$ such that the solution is $(\lceil M(15\sqrt{2}+4)/\varepsilon \rceil, 36, \delta)$-portal-respecting in $D(\boldsymbol{a})$
    transform (π_1, π_2) into an induced two-tour solution Π
    $\mathrm{Sol} \leftarrow \mathrm{Sol} \cup \{\Pi\}$
$\Pi' \leftarrow \mathrm{argmin}\{l(\Pi) : \Pi \in \mathrm{Sol}\}$
return $\textsc{Back-Perturbation}(\Pi')$

Online Flexible Busy Time Scheduling on Heterogeneous Machines

ESA 2024

Setting

- A set of unit-length jobs arrive online, and it is denoted J with $|J| = n$. Job $j \in J$ is equipped with integral arrival r_j and deadline d_j.A machine of type $k \in Z_{\geq 0}$ can execute at most $B_k \in Z_{\geq 1}$ many jobs at once and costs $c_k \geq 1$ per time unit.
- There are an unlimited number of machines of type k.
- The goal is to find a non-preemptive schedule completing all jobs by their deadlines with minimum total cost.

Other Settings

- homogenueous machines
- interval jobs
- length of jobs
- something about future

- have some cost to turn on or off

Result

- Lower Bound: The competitive ratio of any deterministic online algorithm for online unit-length busy time scheduling on heterogeneous machines is at least 2.
- There is an 8-competitive algorithm for online unit-length busy time scheduling on heterogeneous machines with run-time $O(nlogn)$, for n the total number of jobs.
- If jobs have $agreeable$ deadlines,, there is a 2-competitive algorithm for online unit-length busy time scheduling on heterogeneous machines with run-time $O(nK + nlogn)$, for n the total number of jobs and K the number of distinct machine types.

good case

Algorithm 1 Greedy

```
Let W ← ∅, 𝒳 ← ∅.
for τ ← 1 to T do
    ▷ Let J′ be the set of jobs with arrival time τ.
    ▷ Update W ← W ∪ J′.
    if ∃ j* ∈ W with d_{j*} = τ then
        ▷ 𝒳_τ ← GetOptimalBatches(W).
        ▷ Execute all batches in 𝒳_τ.   // Thus completing all jobs in W
        ▷ Update 𝒳 ← 𝒳 ∪ 𝒳_τ.
        ▷ W ← ∅.
    end if
end for
return A schedule 𝒳.   // With cost at most 2 ·cost(OPT)
```

Main Technology

Trade-off

- With only losing a factor of 2 in the competitive ratio, we can assume in online unit-length busy time scheduling on heterogeneous machines that for all $k \in Z_{\geq 0}$, $c_k = 2^p$ for some $p \in Z_{\geq 0}$, and c_0 = 1.

Proposition:
without loss of generality the cost-per job is non-increasing in the machine types, i.e., $\frac{c_0}{B_0} \geq \frac{c_1}{B_1} \geq \ldots$.

Tool&ALG

Definition 9 *For a set of jobs J equipped with arrival times and deadlines in $[0,T]$, let an* interval assignment *$\mathcal{A}$ be a family of tuples $\{(I, t_I, J_I)\}$ of a continuous interval $I \subseteq [0,T]$, a type t_I, and a set of jobs $J_I \subseteq J$. Let $\mathcal{L}$ denote the multi-set of all intervals I with $(I, t_I, J_I) \in \mathcal{A}$ and let $I(j)$ be the interval that job j is assigned to. We define a* valid interval assignment *as one such that the following holds:*

1. *When $t_I \geq 1$, then $|J_I| = B_{t_I - 1}$, and when $t_I = 0$, then $|J_I| = 1$.*
2. *If job j is in J_I, then $[r_j, d_j] \subseteq I$.*
3. *For any two intervals $I, I' \in \mathcal{L}$, if $t_I = t_{I'}$, then I and I' are disjoint.*
4. *Every job is assigned to at most one J_I. If a job j is not assigned to any J_I, then $I(j) = \emptyset$.*

For any valid $\mathcal{L}$:

$$cost(OPT) \geq \frac{1}{4} \sum_{I \in \mathcal{L}} 2^{t_I}$$

Algorithm 2 The main algorithm

Let $W \leftarrow \emptyset$, $\tau \leftarrow 0$, and $\mathcal{X} \leftarrow \emptyset$.
while $\tau \neq \mathsf{NULL}$ **do**
  $\triangleright$ Let J' be the set of jobs with arrival time τ.
  $\triangleright$ Update $W \leftarrow W \cup J'$.
  while $\exists\, j^* \in W$ with $d_{j^*} = \tau$ **do** // *If many candidates, choose j^* arbitrarily from them*
    $\triangleright$ Run Algorithm 3 with inputs τ, J, W, j^*, and $\mathcal{X}$, which outputs new batch X^*, equipped with $t(X^*)$, $J(X^*), \tau(X^*), \widetilde{S}(X^*)$, $\widetilde{I}(X^*)$.
    $\triangleright$ Update $\mathcal{X} \leftarrow \mathcal{X} \cup X^*$ and $W \leftarrow W \setminus J(X^*)$.
    $\triangleright$ $\{(\widetilde{I}(X), t(X), \widetilde{S}(X))\}_{X \in \mathcal{X}} \leftarrow \{(\widetilde{I}(X), t(X), \widetilde{S}(X))\}_{X \in \mathcal{X}} \cup \{(\widetilde{I}(X^*), t(X^*), \widetilde{S}(X^*)\}$.
  end while
  Increment $\tau \leftarrow \tau + 1$, or let $\tau = \mathsf{NULL}$ if time has ended.
end while
Return $\mathcal{X}$.

Algorithm 3 Deciding the next batch

Input: time τ, jobs J, waiting jobs W, critical job j^* with $d_{j^*} = \tau$, and batches $\mathcal{X}$, where every $X \in \mathcal{X}$ is equipped with $t(X)$, $J(X)$, $\tau(X)$, $\widetilde{S}(X)$, $\widetilde{I}(X)$.

$\triangleright$ Let $k \leftarrow 0$.

$\triangleright$ Let $I_0 \leftarrow [r_{j^*}, \tau]$.

while I_k contains a time slot $\tau_k := \tau(X_k)$ with $t(X_k) = k$ for some $X_k \in \mathcal{X}$ **do**

// *Only the latest such* τ_k *is stored*

  $\triangleright$ Let j_k be a job in $J(X_k)$ with the earliest arrival time. // *Tie-break arbitrarily*

  $\triangleright$ Let τ'_k be the arrival time of j_k.

  $\triangleright$ Set $I_{k+1} \leftarrow I_k \cup [\tau'_k, \tau_k]$.

  $\triangleright$ Increment $k \leftarrow k + 1$.

end while

Set $I^* \leftarrow I_k$ and $S^* \leftarrow \{j^*\}$.

if $k > 0$: **then**

  Add all the non-critical jobs of X_{k-1} to S^*.

end if

$\triangleright$ Let J^* be the B_k ($|W|$ if $|W| < B_k$) jobs in W with earliest deadline. // *Tie-break arbitrarily.*

$\triangleright$ Create batch X^* with type $t(X^*) \leftarrow k$, execution time $\tau(X^*) \leftarrow \tau$, and jobs $J(X^*) \leftarrow J^*$.

$\triangleright$ Define $\widetilde{I}(X^*) \leftarrow I^*$ and $\widetilde{S}(X^*) \leftarrow S^*$.

Return X^*.

Appendix