

University of Iceland

Faculty of Industrial Eng., Mechanical Eng. and Computer Science

ALIC

Helg

Introduction

Problem space

Subspace of instances

Foature space

•

Performance

space

instance space

Preference s

learning

onclusion

ALICE

Analysis & Learning Iterative Consecutive Executions

Helga Ingimundardóttir

University of Iceland

June 30, 2016



Motivation:

* The general goal is to train optimisation algorithms, for an arbitrary problem domain, using data.



Introduction

ALIC

Helg

Introduction

space Subspace c

instances

Algorithm

Performance space

Footprints in instance space

Preference s

learning

Conclusions

Motivation:

* The general goal is to train optimisation algorithms, for an arbitrary problem domain, using data.

Contribution:

* The main contribution of this thesis is towards a better understanding of how this training data should be constructed.



ALIC

Heig

Introduction

Problem space

Subspace of instances

Instances

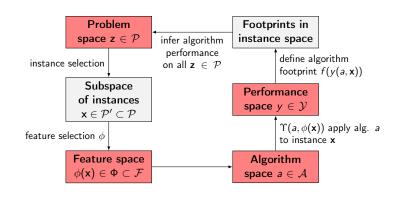
Algorithm

Performance space

Footprints in instance space

Preference so

Preference learning





ALIC

Heig

Introduction

Problem space

Subspace of instances

instances

Algorithm

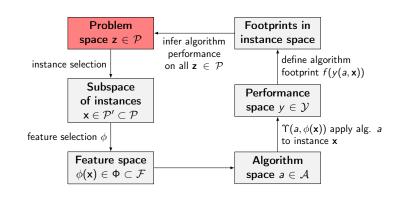
Performance space

Footprints in instance space

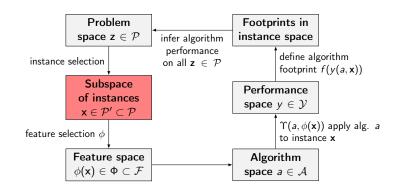
Preference se

Preference

. . .









ALIC

Helg

Introduction

Problem space

Subspace of instances

instances

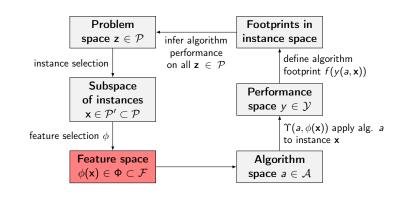
Algorithm

Performance space

Footprints in instance space

Preference se

Preference learning





ALIC

Helg

Introduction

Problem space

Subspace of instances

- .

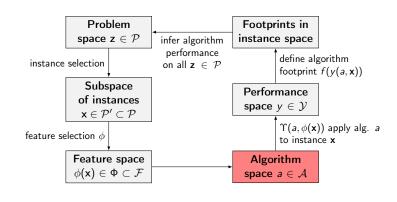
Algorithm

Performance space

Footprints in instance space

Preference se

Preferen





ALIC

Helg

Introduction

Problem space

Subspace of instances

instances

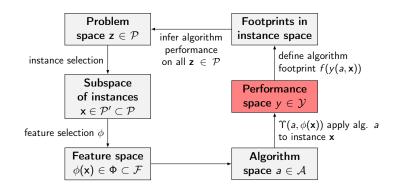
Algorithm

Performance space

Footprints in instance space

Preference se

Preference





ALIC

Heig

Introduction

Problem

Subspace of

instances

Algorithm

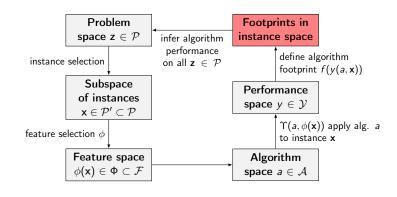
Performance space

Footprints in instance space

Preference se

Preference

. . .





ALIC

Helg

Introduction

Problem

Subspace of instances

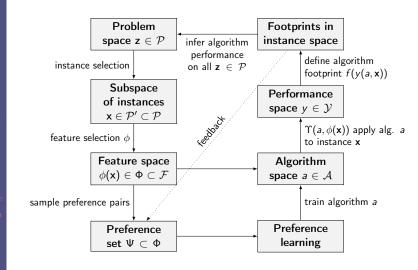
Footure coa

Algorithm

Performance space

Footprints in instance space

Preference learning





ALIC

Heig

Introduction

Problem

Subspace of

_ .

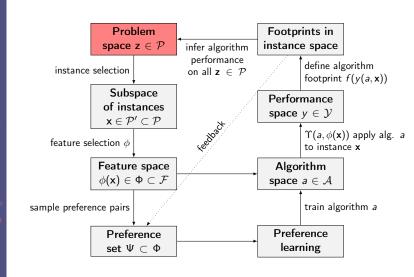
Algorithm

Performance space

Footprints in instance spac

Preference s Preference

learning





ALIC

Helg

Introduction

Problem

Subspace of

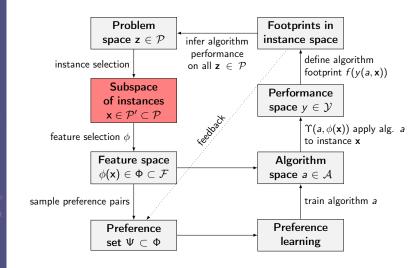
Footuus one

Algorithm

Performance space

instance spac

Preference learning





ALIC

Helg

Introduction

Problem

Subspace of

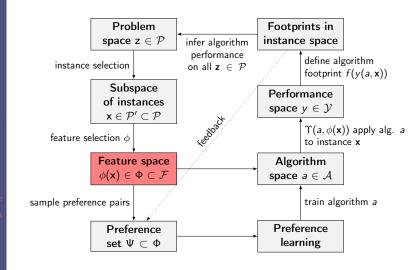
- .

Algorithm

Performance space

instance spac Preference se

Preferen





ALIC

Heig

Introduction

Problem

Subspace of

instances

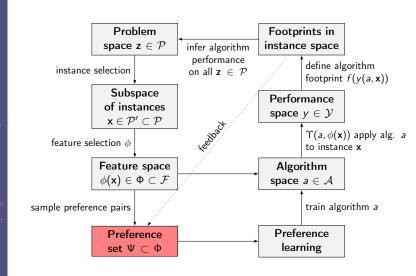
Algorithm

Performance space

Footprints in instance spac

Preterence s

Preferer learning





ALIC

Helg

Introduction

Problem

Subspace of

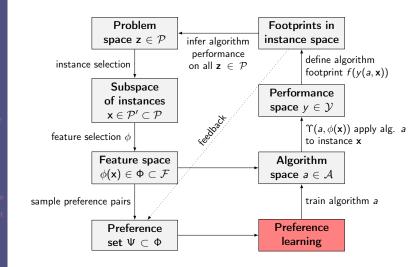
- .

Algorithm

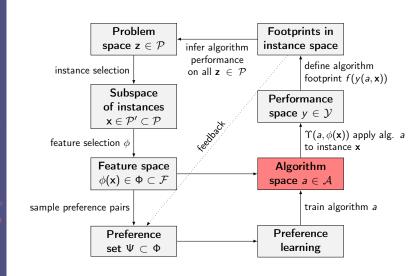
Performance space

instance space

Prefere learning









ALIC

Helg

Introduction

Problem

Subspace of instances

instances

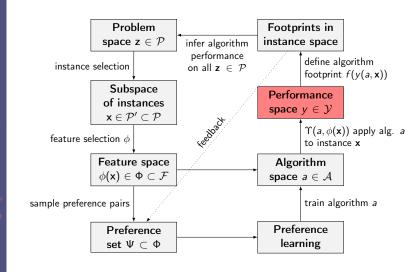
Algorithm

Performance space

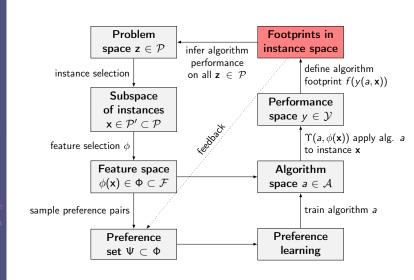
Footprints in instance space

Preference si Preference

Preferer learning









ALIC

Helg

Introduction

Problem space

Subspace of instances

Feature sna

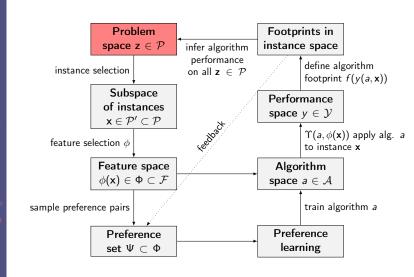
Algorithm

Performance space

Footprints in instance space

Preference so

Preferei learning





ALIC

Helg

Introductio

Problen space

Subspace of

Feature spa

Algorithm

Performan

Footprints in instance space

Preference s

Preferer learning





The attending guests: They all have to:

 J_1) Alice M_1) have wine or pour tea

 J_2) March Hare M_2) spread butter

 M_3) get a haircut J_3) Dormouse

 J_4) Mad Hatter. M_4) check the time of the broken watch

 M_5) say what they mean.

This can be considered as is a typical 4×5 job-shop, where:

- \star objective is to minimise C_{max} (when Alice can leave).



ALIC

Helg

Introducti P<mark>roblem</mark>

Subspace o

Feature spar Algorithm Space

Performance space

Preference se

earning

The attending guests: They all have to:

 J_1) Alice M_1) have wine or pour tea

 J_2) March Hare M_2) spread butter

 J_3) Dormouse M_3) get a haircut

 M_4) Mad Hatter. M_4) check the time of the broken watch

 M_5) say what they mean.

This can be considered as is a typical 4×5 job-shop, where:

- ⋆ our guests are the jobs
- * their tasks are the machine
- \star objective is to minimise C_{max} (when Alice can leave).



ALIC

Helg

Problem

Subspace of instances

lgorithm pace

Performance space

Preference earning The attending guests: They all have to:

 J_1) Alice M_1) have wine or pour tea

 J_2) March Hare M_2) spread butter

 J_3) Dormouse M_3) get a haircut

 J_4) Mad Hatter. M_4) check the time of the broken watch

 M_5) say what they mean.

This can be considered as is a typical 4×5 job-shop, where:

★ our guests are the jobs

* their tasks are the machines

* objective is to minimise C_{max} (when Alice can leave).



ALIC

Helg

Proble

Subspace of instances

algorithm pace

pace Footprints in

Preference earning The attending guests: They all have to:

 J_1) Alice M_1) have wine or pour tea

 J_2) March Hare M_2) spread butter

 J_3) Dormouse M_3) get a haircut

 J_4) Mad Hatter. M_4) check the time of the broken watch

 M_5) say what they mean.

This can be considered as is a typical 4×5 job-shop, where:

★ our guests are the jobs

* their tasks are the machines

 \star objective is to minimise C_{max} (when Alice can leave).



Mad Hatter tea-party (k-solutions)

ALIC

Helg

Introduction

Problem

Subspace of instances

at 'd

Algorithm snace

Performance space

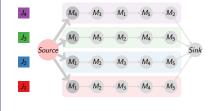
Footprints in instance spa

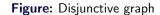
Preference s

Preteren learning

. maluaiam

Start: k = 0





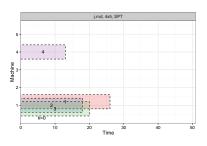


Figure: Gantt chart



Mad Hatter tea-party (k-solutions)

ALIC

Helg

Introduction

Problem

Subspace of instances

Al.....'Al....

Algorithm space

Performance space

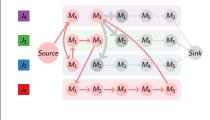
Footprints in instance space

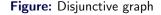
Preference so

learning

anclusion

Midway: k = 10





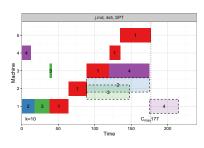


Figure: Gantt chart



Mad Hatter tea-party (k-solutions)

ALIC

Helg

Introduction

Problem space

Subspace of instances

Feature spa

Algorithm

Performance

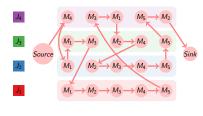
Footprints in instance space

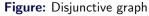
Preference se

Prefere

onclusion

Finish: k = 20





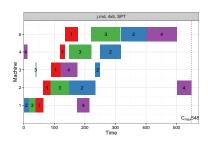


Figure: Gantt chart



Mad Hatter tea-party (K-solutions)

ALIC

Helga

Introduction

Problem

Subspace of instances

Feature spa

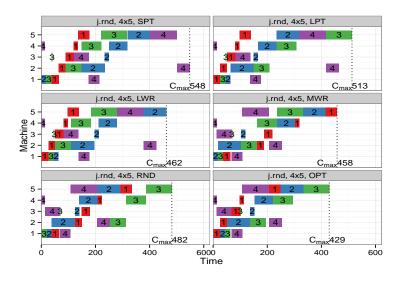
Algorithm

Performanc space

Footprints in instance space

Preference s

Preferen





ALIC

Helg

Introduction Problem space

Subspace of instances

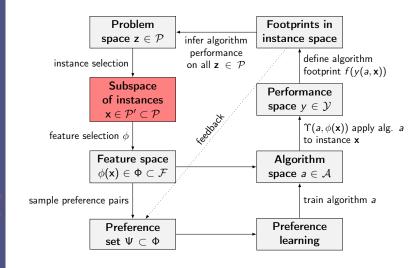
_

Algorithm

Performance space

instance space

Prefere





Problem instance generators

ALIC

Helga

ntroduction Problem

Subspace of instances

Feature spac

Algorithm

Performanc space

Footprints in instance space

Preference s

learning

	name	size $(n \times m)$	N_{train}	N_{test}	note
	$\mathcal{P}_{i.rnd}^{6 \times 5}$	6 × 5	500	500	random
	$\mathcal{P}_{j.rndn}^{6 \times 5}$	6×5	500	500	random-narrow
	$\mathcal{P}_{i.rnd,J_1}^{6\times 5}$	6×5	500	500	random with job variation
_	$\mathcal{P}_{i.rnd,M_1}^{6\times5}$	6×5	500	500	random with machine variation
JSP	$\mathcal{P}_{j.rnd}^{10 \times 10}$	10×10	300	200	random
	$\mathcal{P}_{j.rndn}^{10 imes 10}$	10×10	300	200	random-narrow
	$\mathcal{P}_{i.rnd,J_1}^{10\times10}$	10×10	300	200	random with job variation
	$\mathcal{P}_{i.rnd,M_1}^{10\times 10}$	10×10	300	200	random with machine variation
	$\mathcal{P}_{\mathit{JSP.ORLIB}}$	various	_	82	various
	$\mathcal{P}_{f,rnd}^{6\times5}$	6 × 5	500	500	random
	$\mathcal{P}_{f,rndn}^{6\times5}$	6×5	500	500	random-narrow
_	$\mathcal{P}_{f,ic}^{6\times5}$	6×5	500	500	job-correlated
FSP	$\mathcal{P}_{f,mc}^{6\times5}$	6×5	500	500	machine-correlated
	$\mathcal{P}_{f.mxc}^{6\times5}$	6×5	500	500	mixed-correlation
	$\mathcal{P}_{f,rnd}^{10\times10}$	10×10	300	200	random
	$\mathcal{P}_{\textit{FPS.ORLIB}}$	various	_	31	various



ALIC

Helg

ntroduction Problem

Subspace o

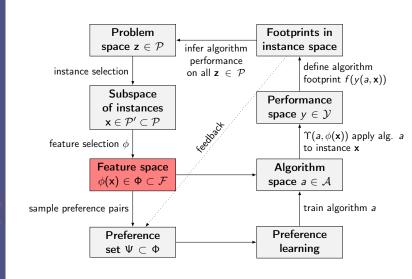
Feature space

Performance

Footprints in instance spac

Preference s

Preterer learning





doį	$egin{array}{c} \phi_1 \ \phi_2 \ \phi_3 \ \phi_4 \ \phi_5 \ \phi_6 \ \phi_7 \ \phi_8 \end{array}$	job processing time job start-time job end-time job arrival time time job had to wait total processing time for job total work remaining for job number of assigned operations for job
machine	$egin{array}{c} \phi_9 \ \phi_{10} \ \phi_{11} \ \phi_{12} \ \phi_{13} \ \phi_{14} \ \phi_{15} \ \phi_{16} \ \end{array}$	when machine is next free total processing time for machine total work remaining for machine number of assigned operations for machine change in idle time by assignment total idle time for machine total idle time for all machines current makespan
final makespan	ϕ_{17} ϕ_{18} ϕ_{19} ϕ_{20} ϕ_{RND} ϕ_{21} ϕ_{22} ϕ_{23} ϕ_{24}	final makespan using SPT final makespan using LPT final makespan using LWR final makespan using MWR final makespans using 100 random rollouts mean for $\phi_{\rm RND}$ standard deviation for $\phi_{\rm RND}$ minimum value for $\phi_{\rm RND}$ maximum value for $\phi_{\rm RND}$



doį	φ1 φ2 φ3 φ4 φ5 φ6 φ7 φ8	job processing time job start-time job end-time job arrival time time job had to wait total processing time for job total work remaining for job number of assigned operations for job
machine	$egin{array}{c} \phi_9 \ \phi_{10} \ \phi_{11} \ \phi_{12} \ \phi_{13} \ \phi_{14} \ \phi_{15} \ \phi_{16} \ \end{array}$	when machine is next free total processing time for machine total work remaining for machine number of assigned operations for machine change in idle time by assignment total idle time for machine total idle time for all machines current makespan
final makespan	ϕ_{17} ϕ_{18} ϕ_{19} ϕ_{20} ϕ_{RND} ϕ_{21} ϕ_{22} ϕ_{23} ϕ_{24}	final makespan using SPT final makespan using LPT final makespan using LWR final makespan using MWR final makespans using 100 random rollouts mean for $\phi_{\rm RND}$ standard deviation for $\phi_{\rm RND}$ minimum value for $\phi_{\rm RND}$ maximum value for $\phi_{\rm RND}$



Features for JSF

ALIC

Helg

Problem

Subspace o instances

-eature space

Algorithm

Performance space

Footprints in instance spac

Preference earning

earning

		<u>jo</u> p	
		. –	

doį	$egin{array}{c} \phi_1 \ \phi_2 \ \phi_3 \ \phi_4 \ \phi_5 \ \phi_6 \ \phi_7 \ \phi_8 \ \end{array}$	job processing time job start-time job end-time job arrival time time job had to wait total processing time for job total work remaining for job number of assigned operations for job
machine	$\phi_9 \ \phi_{10} \ \phi_{11} \ \phi_{12} \ \phi_{13} \ \phi_{14} \ \phi_{15} \ \phi_{16}$	when machine is next free total processing time for machine total work remaining for machine number of assigned operations for machine change in idle time by assignment total idle time for machine total idle time for all machines current makespan
final makespan	ϕ_{17} ϕ_{18} ϕ_{19} ϕ_{20} ϕ_{RND} ϕ_{21} ϕ_{22} ϕ_{23} ϕ_{24}	final makespan using SPT final makespan using LPT final makespan using LWR final makespan using MWR final makespans using 100 random rollouts mean for $\phi_{\rm RND}$ standard deviation for $\phi_{\rm RND}$ minimum value for $\phi_{\rm RND}$ maximum value for $\phi_{\rm RND}$



Trajectory strategies for 4

ALIC

Heig

Introduction

Problem space

Subspace o instances

Footure on

Algorithm

Performance space

Footprints in instance space

Preference se

Prefere

. . .

Following the policy:

- \star (Φ^{OPT}) expert π_{\star} .
- * (Φ^{SPT}) shortest processing time (SPT).
- * (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- * (Φ^{MWR}) most work remaining (MWR).
- \star (Φ ^{RND}) random policy (RND).
- \star ($\Phi^{ES.\rho}$) the policy obtained by optimising with CMA-ES.
- \star (Φ^{ALL}) union of all of the above.



Trajectory strategies for 4

ALIC

пец

Introduction

space

bubspace of instances

Feature space

.

Performance

Footprints in instance space

Preference se

Prefere

C----

- \star (Φ^{OPT}) expert π_{\star} .
- * (Φ^{SPT}) shortest processing time (SPT).
- * (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- \star (Φ^{MWR}) most work remaining (MWR).
- \star (Φ ^{RND}) random policy (RND).
- \star ($\Phi^{\text{ES},\rho}$) the policy obtained by optimising with CMA-ES.
- \star (Φ^{ALL}) union of all of the above.



Trajectory strategies for \$\psi\$

ALIC

пец

Introduction

space

Subspace of instances

Feature space

Performance

Footprints in instance space

Preference s

Prefere

Canalusian

- \star (Φ^{OPT}) expert π_{\star} .
- \star (Φ^{SPT}) shortest processing time (SPT).
- * (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- \star (Φ^{MWR}) most work remaining (MWR).
- \star (Φ^{RND}) random policy (RND).
- \star ($\Phi^{\text{ES},\rho}$) the policy obtained by optimising with CMA-ES.
- \star (Φ^{ALL}) union of all of the above.



- * (Φ^{OPT}) expert π_* .
- \star (Φ^{SPT}) shortest processing time (SPT).
- \star (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- \star (Φ^{MWR}) most work remaining (MWR).
- \star (Φ ^{RND}) random policy (RND).
- \star ($\Phi^{\text{ES},\rho}$) the policy obtained by optimising with CMA-ES.
- \star (Φ^{ALL}) union of all of the above.



- * (Φ^{OPT}) expert π_* .
- \star (Φ^{SPT}) shortest processing time (SPT).
- \star (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- \star (Φ^{MWR}) most work remaining (MWR).
- \star (Φ ^{RND}) random policy (RND).
- \star ($\Phi^{\text{ES},\rho}$) the policy obtained by optimising with CMA-ES.
- \star (Φ^{ALL}) union of all of the above.



- * (Φ^{OPT}) expert π_* .
- \star (Φ^{SPT}) shortest processing time (SPT).
- \star (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- \star (Φ^{MWR}) most work remaining (MWR).
- \star (Φ ^{RND}) random policy (RND).
- \star ($\Phi^{\text{ES},\rho}$) the policy obtained by optimising with CMA-ES.
- \star (Φ^{ALL}) union of all of the above.



Trajectory strategies for Φ

ALIC

Helg

Introductio Problem space

Subspace o instances

eature space

Moorithm

Performance space

instance spac Preference se

learning

onclusion:

- \star (Φ^{OPT}) expert π_{\star} .
- \star (Φ^{SPT}) shortest processing time (SPT).
- \star (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- \star (Φ^{MWR}) most work remaining (MWR).
- \star (Φ^{RND}) random policy (RND).
- * $(\Phi^{ES.\rho})$ the policy obtained by optimising with CMA-ES.
- \star (Φ^{ALL}) union of all of the above.



- \star (Φ^{OPT}) expert π_{\star} .
- \star (Φ^{SPT}) shortest processing time (SPT).
- \star (Φ^{LPT}) longest processing time (LPT).
- \star (Φ^{LWR}) least work remaining (LWR).
- \star (Φ^{MWR}) most work remaining (MWR).
- \star (Φ ^{RND}) random policy (RND).
- \star ($\Phi^{ES.\rho}$) the policy obtained by optimising with CMA-ES.
- \star (Φ ^{ALL}) union of all of the above.



Sampled size of $|\Phi(k)|$ $(6 \times 5, N_{toin} = 500)$

ALIC

Helg

Introduction

Problem space

Subspace o

Feature space

Algorithm

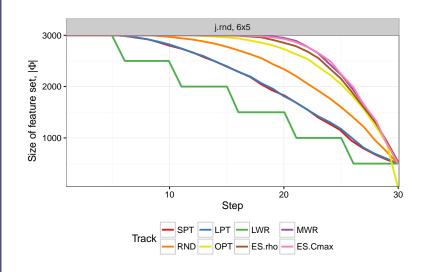
Performanc

Footprints in instance spa

Preference s

_ _ _

learning





Framework for Algorithm Learning

ALIC

Helg

Introduction Problem

Subspace of instances

Algorithm

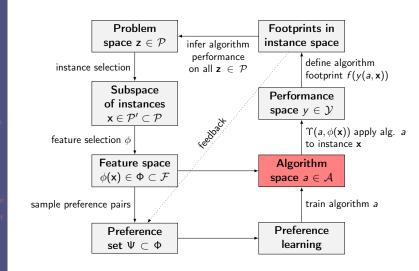
Performanc

space

instance spac Preference se

Preference learning

onclusions





Various Methods for Solving JSP

ALIC

Helg

Introduction

Problem space

Subspace of instances

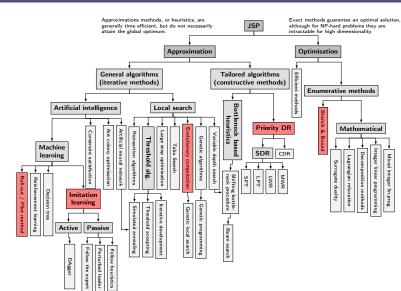
Algorithm

Performanc space

Footprints in instance space

Preference s

Prefere





Various Methods for Solving JSP

ALIC

Helg

Introduction

Problem space

Subspace of instances

Algorithm

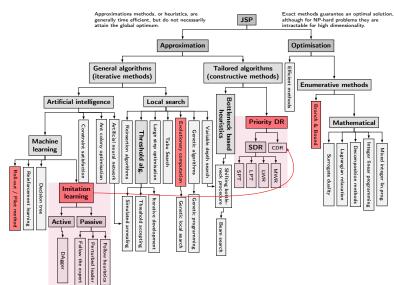
space

space

Preference se

Preference :

learning





Framework for Algorithm Learning

ALIC

Helg

Introduction Problem

Subspace of

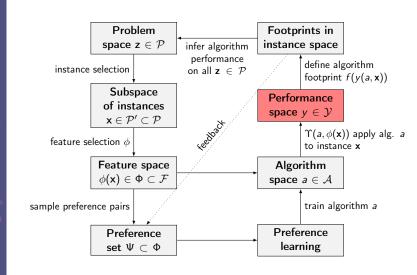
Feature space
Algorithm

Performance space

Preference se

learning

onclusions





Performance measure

ALICI

Helg

Introduction

Subspace of

nstances

Algorithm space

Performance space

Footprints in instance space

Preference s

Prefere learning

onclusion

Performance of policy π compared with its optimal makespan, found using an expert policy, π_{\star} , is the following loss function:

$$\rho = \frac{C_{\mathsf{max}}^{\pi} - C_{\mathsf{max}}^{\pi_{\star}}}{C_{\mathsf{max}}^{\pi_{\star}}} \cdot 100\%$$

The goal is to minimise this discrepancy between predicted value and true outcome.



Framework for Algorithm Learning

ALIC

Helg

Introduction -- · ·

space

Subspace of instances

Feature spa

Algorithm space

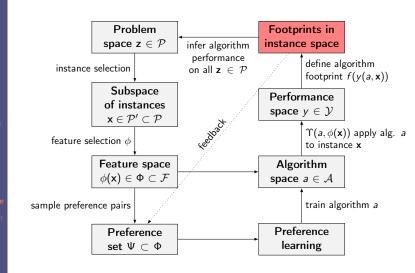
Performance space

Footprints in instance space

Preference se

Prefere learning

onclusions





Deviation from optimality, /

ALIC

Helga

Introduction

Problem space

instances

Feature spa

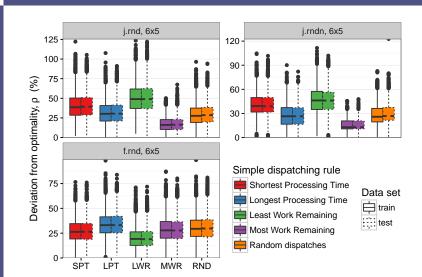
Algorithm space

Pertormance space

instance space

Preference s

Preferent learning





Deviation from optimality, /

ALIC

Helga

Introductio Problem

Subspace of

Feature spa

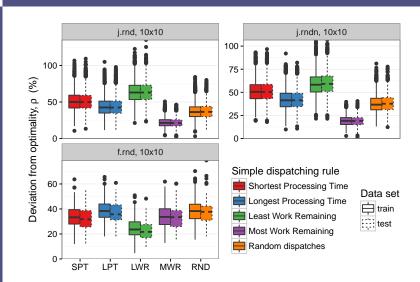
Algorithm space

Pertormance space

Footprints in instance space

Preference so

Preterer learning





Making optimal decisions, ξ

ALIC

Helg

Introduction

Problem space

Subspace o instances

Feature spa

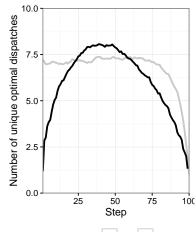
Algorithm space

Performance space

Footprints in instance space

Preference se

......





Making optimal decisions, ξ

ALIC

Helg

Introduct Problem

Subspace of instances

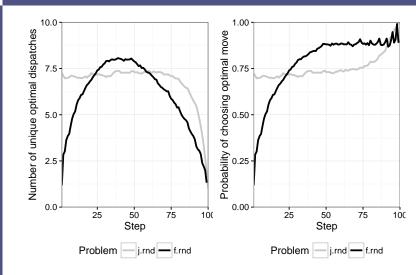
Algorithm
space

Performance space

Footprints in instance space

Preference learning

`onclusio





Probability of SDR being optimal, $\xi_{\langle \text{SDR}}^{\star}$

ALICE

Helg

Introductio

Problem

Subspace of

instances

Algorithm

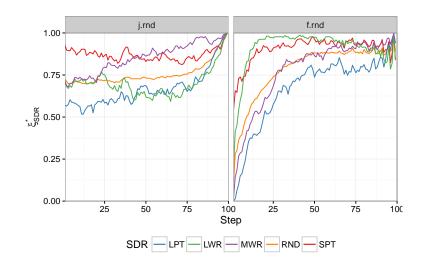
space

space

Footprints in instance space

Preference s

Prefere





Blended dispatching rule

ALIC

Helga

ntroduction

Problem space

Subspace o instances

Feature spa

Algorithm

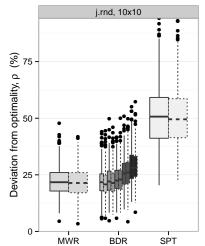
Performance space

Footprints in instance space

Preference so

Prefere learning

onclusion



Dispatching rule

Shortest Processing Time

Most Work Remaining

SPT (first 10 %), MWR (last 90 %)

SPT (first 15 %), MWR (last 85 %)

SPT (first 20 %), MWR (last 80 %)

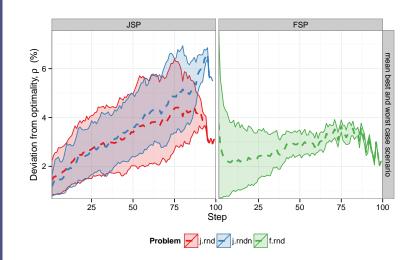
SPT (first 30 %), MWR (last 70 %)

SPT (first 40 %), MWR (last 60 %)

Data set

train 🚃 test







Probability of SDR being optimal, $\xi_{(SDR)}$

ALIC

Helg

Introduction

Problem

Subspace of

Feature spa

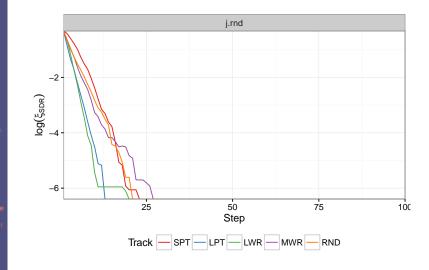
Algorithm

Performance space

Footprints in instance space

Preference s

Prefere





Impact of suboptimal decision, $\{\zeta_{\min}^{\pi}, \zeta_{\max}^{\pi}\}$

ALICI

Helga

Introduction

Problem

Subspace of

instances

Algorithm

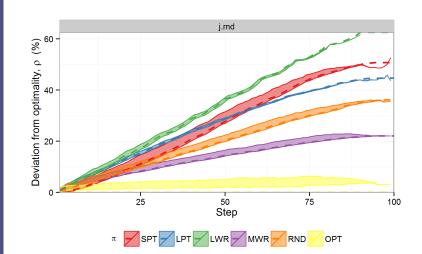
Algorithm

Performance space

Footprints in instance space

Preference s

Preferer learning





Framework for Algorithm Learning

ALIC

Helg

ntroductio Problem

Subspace of instances

_ .

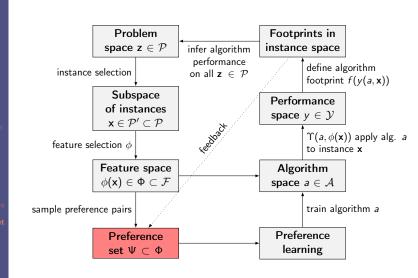
Algorithm

Performance space

Footprints in instance spa

Preference se

Preferen





ALICE framework for creating dispatching rules:

- * Linear classification to identify good dispatches, from worse ones.

- \star Sample Φ to create training set Ψ with rank pairs:
 - \star optimal decision, $(z^{\circ}, y_{\circ}) = (\phi^{\circ} \phi^{\circ}, +1)$
 - * suboptimal decision, $(\mathbf{z}^s, y_s) = (\phi^s \phi^o, -1)$

 \star Sample Ψ using stepwise bias for time independent policy.



Generating training data

ALIC

Helg

Introduction Problem

Subspace of instances

Algorithm

Performanco space

instance spac

Preference se

learning

ALICE framework for creating dispatching rules:

- ★ Linear classification to identify good dispatches, from worse ones.
- ★ Generate feature set, $\Phi \subset \mathcal{F}$, both from
 - \star optimal solutions, ϕ^o
 - \star suboptimal solutions, ϕ^s

by exploring various trajectories within the feature-space (where $\phi^o, \phi^s \in \mathcal{F}$).

- \star Sample Φ to create training set Ψ with rank pairs:
 - \star optimal decision, $(\mathbf{z}^o, y_o) = (\phi^o \phi^s, +1)$
 - \star suboptimal decision, $(z^s, y_s) = (\phi^s \phi^o, -1)$

using different ranking schemes (where $\mathbf{z}^o,\mathbf{z}^s\in\Psi)$

 \star Sample Ψ using stepwise bias for time independent policy.



Generating training data

ALICE

Helg

Problem space

nstances

lgorithm

Performance space

instance spac

Preference learning ALICE framework for creating dispatching rules:

- * Linear classification to identify good dispatches, from worse ones.
- \star Generate feature set, $\Phi \subset \mathcal{F},$ both from
 - \star optimal solutions, ϕ^o
 - \star suboptimal solutions, ϕ^s

by exploring various trajectories within the feature-space (where $\phi^o, \phi^s \in \mathcal{F}$).

- \star Sample Φ to create training set Ψ with rank pairs:
 - \star optimal decision, $(\mathbf{z}^o, y_o) = (\phi^o \phi^s, +1)$
 - * suboptimal decision, $(\mathbf{z}^s, y_s) = (\phi^s \phi^o, -1)$
 - using different ranking schemes (where $\mathbf{z}^o,\mathbf{z}^s\in\Psi)$
- \star Sample Ψ using stepwise bias for time independent policy.



Generating training data

ALICE

Helg

Problem space

nstances

lgorithm pace

space Footprints in Instance space

Preference se

Preference earning
Conclusions

ALICE framework for creating dispatching rules:

- * Linear classification to identify good dispatches, from worse ones.
- \star Generate feature set, $\Phi \subset \mathcal{F},$ both from
 - \star optimal solutions, ϕ^o
 - \star suboptimal solutions, ϕ^s

by exploring various trajectories within the feature-space (where $\phi^o, \phi^s \in \mathcal{F}$).

- \star Sample Φ to create training set Ψ with rank pairs:
 - \star optimal decision, $(\mathbf{z}^o, y_o) = (\phi^o \phi^s, +1)$
 - * suboptimal decision, $(\mathbf{z}^s, y_s) = (\phi^s \phi^o, -1)$ using different ranking schemes (where $\mathbf{z}^o, \mathbf{z}^s \in \Psi$)
- * Sample Ψ using stepwise bias for time independent policy.



Sampled size of $|\Phi(k)|$ (5 × 5, N_{train} = 500)

ALIC

Helg

Introduction

Problem space

Subspace o instances

reacure spi

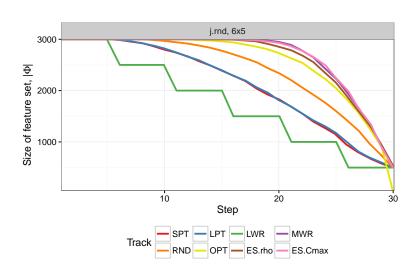
Algorithm

Performanc space

Footprints in instance spa

Preference s

Preference





Sampled size of $|\Psi(k)|$ (6 × 5, $N_{train} = 500$)

ALIC

Helg

Introduction Problem

Subspace o

Feature spa

Algorithm

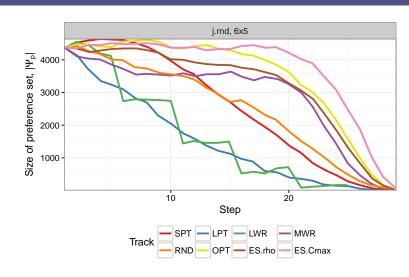
Performanc space

Footprints in instance spa

Preference s

Prefere learnin

onclusio





Stepwise bias strategies

 $(6 \times 5, N_{train} = 500)$

ALIC

Helg

Introduction

Problem space

Subspace of instances

Feature spa

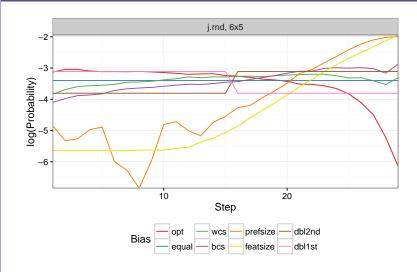
Algorithm

Performance

Footprints in

Preference s

Preference





Framework for Algorithm Learning

ALIC

Helg

ntroductio Problem

Subspace of instances

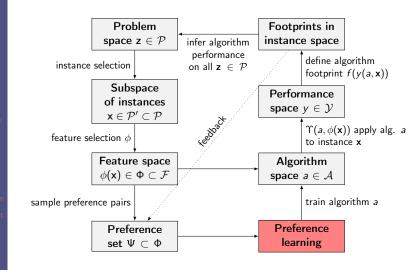
mstances

Algorithm

Performance

Footprints in instance spac

Preference





Ordinal Regression

ALICI

Helg

Introduction

Subspace o

instances

Algorithm

Performance

Footprints in

Preference s

Preference

Conclusion

Preference learning:

* Mapping of points to ranks: $\{h(\cdot): \Phi \mapsto Y\}$ where

$$\phi_o \succ \phi_s \iff h(\phi_o) > h(\phi_s)$$

* The preference is defined by a linear function:

$$h(\phi) = \langle w_i \cdot \phi \rangle$$

optimised w.r.t. w based on training data Ψ

* Note: Limitations in approximation function to capture the complex dynamics incorporated in optimal trajectories



Ordinal Regression

ALIC

Helg

ntroduction

Subspace o

Feature spa

Algorithm

Performance space

Footprints in instance space

Preference learning

Conclusion

Preference learning:

★ Mapping of points to ranks: $\{h(\cdot) : \Phi \mapsto Y\}$ where

$$\phi_o \succ \phi_s \quad \Longleftrightarrow \quad h(\phi_o) > h(\phi_s)$$

* The preference is defined by a linear function:

$$h(\phi) = \langle \mathbf{w}_i \cdot \phi \rangle$$

optimised w.r.t. w based on training data Ψ

* Note: Limitations in approximation function to capture the complex dynamics incorporated in optimal trajectories



Ordinal Regression

ALIC

Helg

ntroductio Problem

Subspace o nstances

-eature spa Algorithm

Performanco space

Footprints in instance space

Preference learning

onclusion

Preference learning:

* Mapping of points to ranks: $\{h(\cdot): \Phi \mapsto Y\}$ where

$$\phi_o \succ \phi_s \quad \Longleftrightarrow \quad h(\phi_o) > h(\phi_s)$$

* The preference is defined by a linear function:

$$h(\phi) = \langle w_i \cdot \phi \rangle$$

optimised w.r.t. \boldsymbol{w} based on training data $\boldsymbol{\Psi}$

* Note: Limitations in approximation function to capture the complex dynamics incorporated in optimal trajectories.



Various Methods for Solving JSP

ALIC

Helg

Introduction

Problem space

instances

Feature spa

Algorithm space

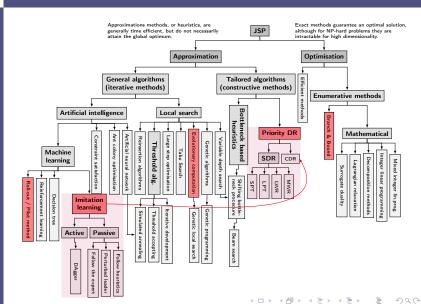
Performance space

Footprints in instance space

Preference :

learning

onclusions





ALIC

Helg

Introduction

Problem

Subspace of

Feature space

A Loro via lo vo

Performanc

Footprints in instance space

Preference s

Preference learning

Conclusio

- * Prediction with expert advice, π_*
- \star Follow the perturbed leader (OPT ϵ)
- * Follow a heuristic (e.g. SDRs).



ALIC

Helg

Introduction

Problem space

Subspace of

Feature snace

A Loronia lono

Performanc

Footprints in

Preference se

Conclusio

- \star Prediction with expert advice, π_{\star}
- \star Follow the perturbed leader (OPT ϵ)
- * Follow a heuristic (e.g. SDRs).



ALIC

Helg

Introduction

Problem space

Subspace of

Feature space

.

· Performano

Footprints in instance space

Preference se

Preference se

Conclusio

- \star Prediction with expert advice, π_{\star}
- \star Follow the perturbed leader (OPT ϵ)
- * Follow a heuristic (e.g. SDRs).



ALIC

Helg

Introduction

Problem space

Subspace of instances

Feature space

Algorithm

Performanc

Footprints in instance space

Preference se

Conclusio

- \star Prediction with expert advice, π_{\star}
- \star Follow the perturbed leader (OPT ϵ)
- * Follow a heuristic (e.g. SDRs).



ALIC

Helg

Introduction Problem Space

Subspace of instances

eature spa Ugorithm pace

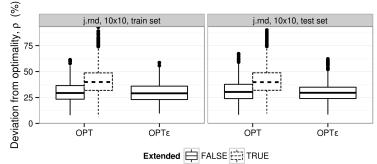
Performance space

Footprints in instance space

Preference learning

Conclusions

- \star Prediction with expert advice, π_{\star}
- \star Follow the perturbed leader (OPT ϵ)
- ★ Follow a heuristic (e.g. SDRs).





ALIC

Helg

Introduction

Problem space

Subspace of instances

Feature space

Algorithm

Performanc

Footprints in

Preference s

Preference

Conclusion

Active imitation learning (iterative):

* Dataset Aggregation (DAgger)

$$\pi_i = \beta_i \pi_* + (1 - \beta_i) \hat{\pi}_{i-1}$$



ALIC

Helg

Introduction

space

Subspace of instances

Feature space

Algorithm

Performance space

Footprints in instance space

Preference se

Droforonco

Conclusio

Active imitation learning (iterative):

⋆ Dataset Aggregation (DAgger)

$$\pi_i = \beta_i \pi_\star + (1 - \beta_i) \hat{\pi}_{i-1}$$



ALIC

Helg

Introduction

Subspace of

instances

reature spac Algorithm

Performanc

Footprints in instance space

Preference se

learning

Conclusio

Active imitation learning (iterative):

⋆ Dataset Aggregation (DAgger)

$$\pi_i = \beta_i \pi_\star + (1 - \beta_i) \hat{\pi}_{i-1}$$



ALIC

Helg

Introduction

Problem

Space

Subspace o instances

Feature spac Algorithm space

Performance space

Footprints in instance space

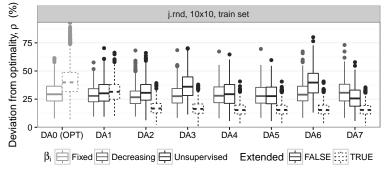
Preference learning

Conclusion

Active imitation learning (iterative):

⋆ Dataset Aggregation (DAgger)

$$\pi_i = \beta_i \pi_\star + (1 - \beta_i) \hat{\pi}_{i-1}$$





Deviation from optimality, /

ALIC

Helg

Introduction

Problem space

instances

r cuture sp

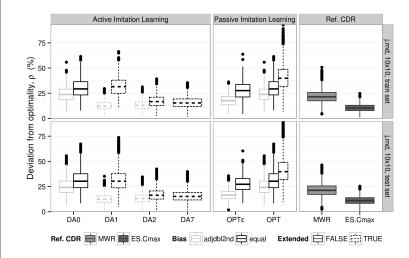
Algorithm

Performanc space

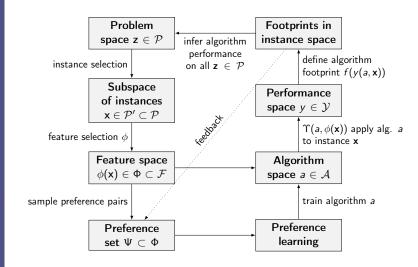
Footprints in instance space

Preference :

Conclusion









ALICE

introduction

°.....

nstances

Algorithm

space Performano

Footprints in instance space

Preference s

learning

Conclusion

The thesis introduced a framework for learning (linear) composite priority dispatching rule – using job-shop as a case-study – with the following guidelines:

- * For a given problem domain, use a suitable problem generator to train and test on.
- \star Define features to grasp the essence of visited k-solutions
- * Success is highly dependent on the preference pairs introduced to the system:
 - $\star \Psi_p$ reduces the preference set without loss of performance.
 - \star Stepwise bias is needed to balance time dependent Ψ_{p} in order to create time independent models.

It is non intuitive how to go about collecting training data.



ALICE

Helg

ntroductio

Subspace of nstances

Algorithm space

Performance space Footprints in

nstance spac Preference se Preference learning The thesis introduced a framework for learning (linear) composite priority dispatching rule – using job-shop as a case-study – with the following guidelines:

- * For a given problem domain, use a suitable problem generator to train and test on.
- \star Define features to grasp the essence of visited k-solutions
- * Success is highly dependent on the preference pairs introduced to the system:
 - $\star \Psi_p$ reduces the preference set without loss of performance.
 - * Stepwise bias is needed to balance time dependent Ψ_{p} in order to create time independent models.

It is non intuitive how to go about collecting training data



ALICE

Helg

ntroductio

Subspace of nstances

Feature spa Algorithm space

Performance space Footprints in nstance spac

Preference earning

The thesis introduced a framework for learning (linear) composite priority dispatching rule – using job-shop as a case-study – with the following guidelines:

- * For a given problem domain, use a suitable problem generator to train and test on.
- \star Define features to grasp the essence of visited k-solutions
- * Success is highly dependent on the preference pairs introduced to the system:
 - $\star \Psi_p$ reduces the preference set without loss of performance.
 - * Stepwise bias is needed to balance time dependent Ψ_{p} in order to create time independent models.

It is non intuitive how to go about collecting training data.



ALICE

Helga

Problem

Subspace of nstances

Feature spac Algorithm Space

Performance space

Footprints in nstance space

Preference set

The thesis introduced a framework for learning (linear) composite priority dispatching rule – using job-shop as a case-study – with the following guidelines:

- * For a given problem domain, use a suitable problem generator to train and test on.
- \star Define features to grasp the essence of visited k-solutions
- * Success is highly dependent on the preference pairs introduced to the system:
 - $\star \Psi_p$ reduces the preference set without loss of performance.
 - * Stepwise bias is needed to balance time dependent Ψ_{p} in order to create time independent models.

It is non intuitive how to go about collecting training data.



ALICE

Helga

Problem
pace
Subspace of

eature spac Algorithm pace Performance pace

pace

cotprints in
nstance space

reference se
earning

conclusions

The thesis introduced a framework for learning (linear) composite priority dispatching rule – using job-shop as a case-study – with the following guidelines:

- * For a given problem domain, use a suitable problem generator to train and test on.
- \star Define features to grasp the essence of visited k-solutions
- * Success is highly dependent on the preference pairs introduced to the system:
 - $\star \Psi_p$ reduces the preference set without loss of performance.
 - * Stepwise bias is needed to balance time dependent Ψ_{p} in order to create time independent models.

It is non intuitive how to go about collecting training data



ALICE

Helga

Problem pace subspace of astances

eature spac algorithm pace erformance

cotprints in istance space Preference set The thesis introduced a framework for learning (linear) composite priority dispatching rule – using job-shop as a case-study – with the following guidelines:

- * For a given problem domain, use a suitable problem generator to train and test on.
- \star Define features to grasp the essence of visited k-solutions
- * Success is highly dependent on the preference pairs introduced to the system:
 - $\star \Psi_p$ reduces the preference set without loss of performance.
 - * Stepwise bias is needed to balance time dependent Ψ_p in order to create time independent models.

It is non intuitive how to go about collecting training data



ALICE

The thesis introduced a framework for learning (linear) composite priority dispatching rule – using job-shop as a case-study – with the following guidelines:

- * For a given problem domain, use a suitable problem generator to train and test on.
- \star Define features to grasp the essence of visited k-solutions
- * Success is highly dependent on the preference pairs introduced to the system:
 - $\star \Psi_p$ reduces the preference set without loss of performance.
 - \star Stepwise bias is needed to balance time dependent Ψ_p in order to create time independent models.

It is non intuitive how to go about collecting training data.



ALICE

Helg

Introduction

Subspace o

instances Feature sp

Algorithm space

Performand space

Footprints in instance space

learning

- \star Learning optimal trajectories predominant in literature. Study showed Φ^{OPT} can result in insufficient knowledge.
- * Following sub-optimal deterministic policies, yet labelling with an optimal solver, improves the guiding policy.
- * Active update procedure using DAgger ensures sample states the learned model is likely to encounter is integrated to $\Psi_p^{\mathrm{DA}i}$.
- * Instead of reusing same problem instances, extend the training set with new instances for quicker convergence of DAgger.
- * In sequential decision making, all future observations are dependent on previous operations.



ALICE

Helg

Introducti Problem space

Subspace of nstances

Feature spac Algorithm space Performance

· Footprints in nstance spac Preference se

reference earning Conclusions

- \star Learning optimal trajectories predominant in literature. Study showed Φ^{OPT} can result in insufficient knowledge.
- ★ Following sub-optimal deterministic policies, yet labelling with an optimal solver, improves the guiding policy.
- * Active update procedure using DAgger ensures sample states the learned model is likely to encounter is integrated to Ψ_p^{DAi} .
- * Instead of reusing same problem instances, extend the training set with new instances for quicker convergence of DAgger.
- * In sequential decision making, all future observations are dependent on previous operations.



ALICE

Helga

ntroduction Problem Space

Subspace of nstances

eature Algorith pace

Performance
pace

ootprints in
nstance spac

reference earning Conclusions

- \star Learning optimal trajectories predominant in literature. Study showed Φ^{OPT} can result in insufficient knowledge.
- * Following sub-optimal deterministic policies, yet labelling with an optimal solver, improves the guiding policy.
- * Active update procedure using DAgger ensures sample states the learned model is likely to encounter is integrated to $\Psi_{p}^{\text{DA}i}$.
- * Instead of reusing same problem instances, extend the training set with new instances for quicker convergence of DAgger.
- * In sequential decision making, all future observations are dependent on previous operations.



ALICE

Helga

Problem pace Jubspace of Instances

digorithm pace Performance pace

Preference so Preference earning

- \star Learning optimal trajectories predominant in literature. Study showed Φ^{OPT} can result in insufficient knowledge.
- * Following sub-optimal deterministic policies, yet labelling with an optimal solver, improves the guiding policy.
- \star Active update procedure using DAgger ensures sample states the learned model is likely to encounter is integrated to $\Psi^{\mathrm{DA}i}_{p}$.
- * Instead of reusing same problem instances, extend the training set with new instances for quicker convergence of DAgger.
- * In sequential decision making, all future observations are dependent on previous operations.



ALICE

Helga

Problem pace Subspace of Instances

Algorithm
pace
Performance
pace

Sootprints in

reference searcherence searning

- \star Learning optimal trajectories predominant in literature. Study showed Φ^{OPT} can result in insufficient knowledge.
- * Following sub-optimal deterministic policies, yet labelling with an optimal solver, improves the guiding policy.
- \star Active update procedure using DAgger ensures sample states the learned model is likely to encounter is integrated to $\Psi^{\mathrm{DA}i}_{p}$.
- * Instead of reusing same problem instances, extend the training set with new instances for quicker convergence of DAgger.
- ★ In sequential decision making, all future observations are dependent on previous operations.



Acknowledgements

ALICI

Helg

ntroductio Problem space Subspace o

Subspace of Instances Feature space

Algorithm pace Performanc pace

instance spa Preference s Preference learning

onclusions

Funding: University of Iceland's Research Fund.

Doctoral committee:

- * Prof. Tómas Philip Rúnarsson, University of Iceland (advisor).
- * Prof. Gunnar Stefánsson, University of Iceland.
- * Prof. Michèle Sebag, Université Paris-Sud.





Thank you for your attention

ALIC

Helg

Problem space

Subspace of nstances

eature spa algorithm pace

Performance space

potprints in estance space

Preference earning

Questions?

Helga Ingimundardóttir hei2@hi.is

Supplementary material:

- * Shiny application
- * Github.

