

Inductive Programming

Lecture 5

Induction of Efficient Programs

Stephen Muggleton
University of Nanjing and
Emeritus Professor Imperial College London

23rd October, 2025

Updated Overview of Inductive Programming

Each **Lecture** immediately followed by **Tutorial**.

Lecture 1	09-10-25	End-user Programming by Induction
Lecture 2	14-10-25	Domain-specific languages and Background Knowledge
Lecture 3	16-10-25	One-shot induction and Bias reformulation
Lecture 4	21-10-25	Inducing an Algorithm from One Example
Lecture 5	23-10-25	Induction of Efficient Programs
Lecture 6	27-10-25	Comprehensibility
Lecture 7	28-10-25	Data wrangling
Lecture 8	30-10-25	Game Strategy Induction

Papers for this lecture

Paper5.1: A. Cropper and S.H. Muggleton. Learning efficient logical robot strategies involving composable objects. In Proceedings of the 24th International Joint Conference Artificial Intelligence (IJCAI 2015), pages 3423-3429. IJCAI, 2015.

Paper5.2: A. Cropper and S.H. Muggleton. Learning efficient logic programs. Machine Learning, 108:1063-1083, 2019.

Motivation

- Inductive Programming
- Few examples per task
- Short programs preferred - Blumer bound
- Are shorter programs always preferable?

Permutation Sort versus Merge Sort

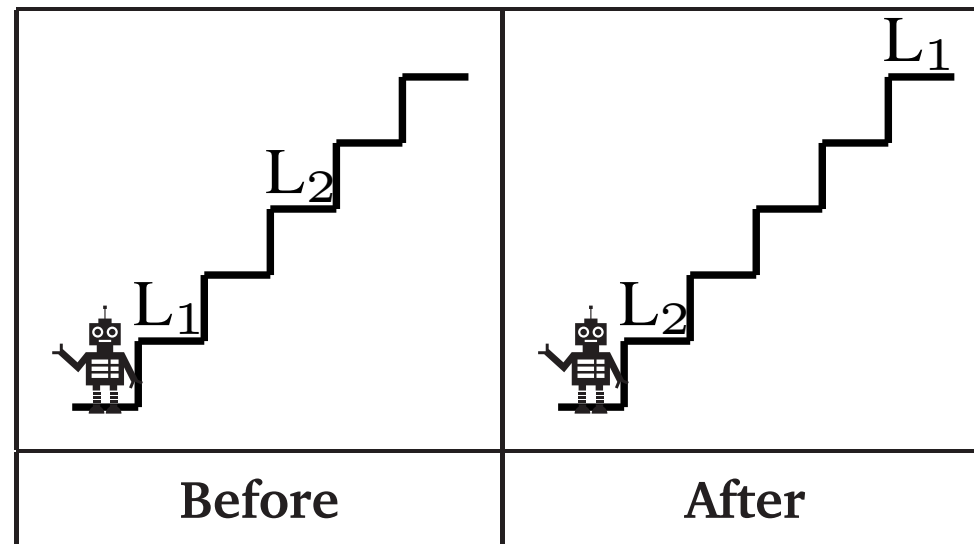
Program size

psort	<code>s(L1,L2) :- permute(L1,L2), sorted(L2).</code>
msort	<code>s([],[]).</code> <code>s(L0,L) :- sp(L0,L1,L2), s(L1,L3), s(L2,L4), m(L3,L4,L).</code>

Time complexity

psort	$O(n!)$
msort	$O(n \log(n))$

Postman [Paper5.1]



n letters and d places for delivery

Postman [Paper5.1]

Metagol_D - $O(nd)$

$p(A,B):- p2(A,C), p(C,B).$

$p(A,B):- p2(A,C), gtb(C,B).$

$p2(A,B):- p1(A,C), gtb(C,B).$

$p1(A,B):- fns(A,C), take(C,B).$

$p1(A,B):- fnr(A,C), give(C,B).$

Metagol_O - $O(n + d)$

$p(A,B):- p2(A,C), p2(C,B).$

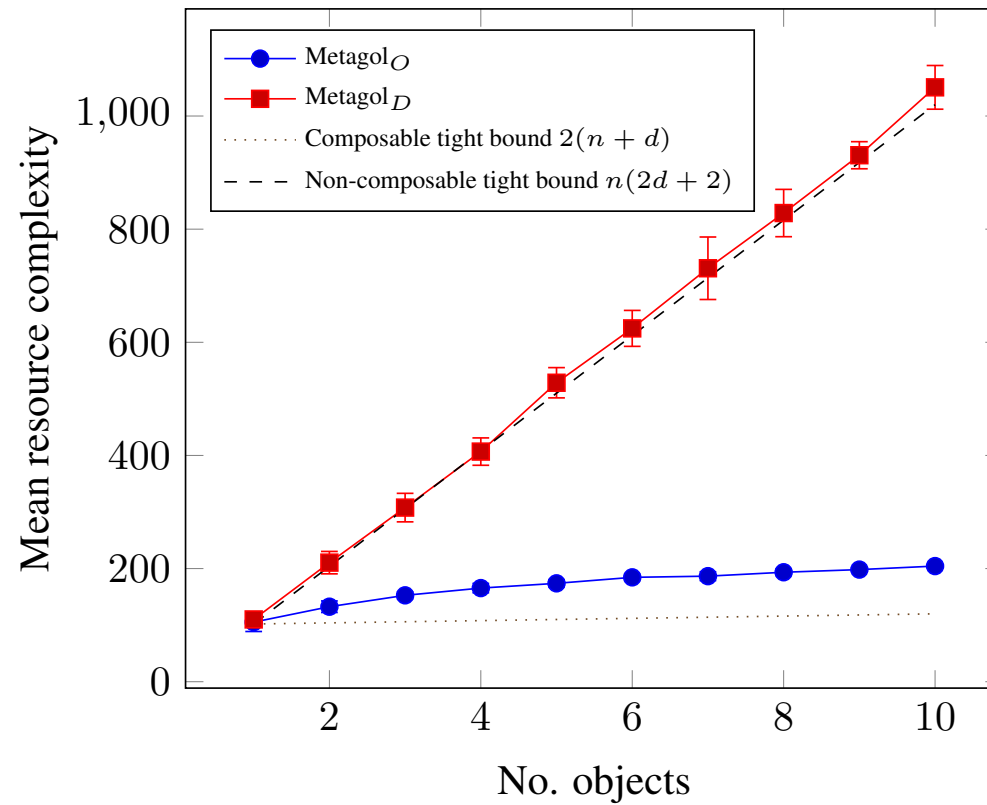
$p2(A,B):- p1(A,C), p2(C,B).$

$p2(A,B):- p1(A,C), gtb(C,B).$

$p1(A,B):- fns(A,C), bag(C,B).$

$p1(A,B):- fnr(A,C), give(C,B).$

Postman mean resource complexity 50 places [Paper5.1]



Robot Letter Sorter [Paper5.1]

Metagol_D - $O(n^2)$

rs(A,B):- rs1(A,C), rs(C,B).

rs1(A,B):- cmp(A,C), rs1(C,B).

rs1(A,B):- dec(A,C), gst(C,B).

rs(A,B):- rs1(A,C), gst(C,B).

Metagol_O - $O(n \log(n))$

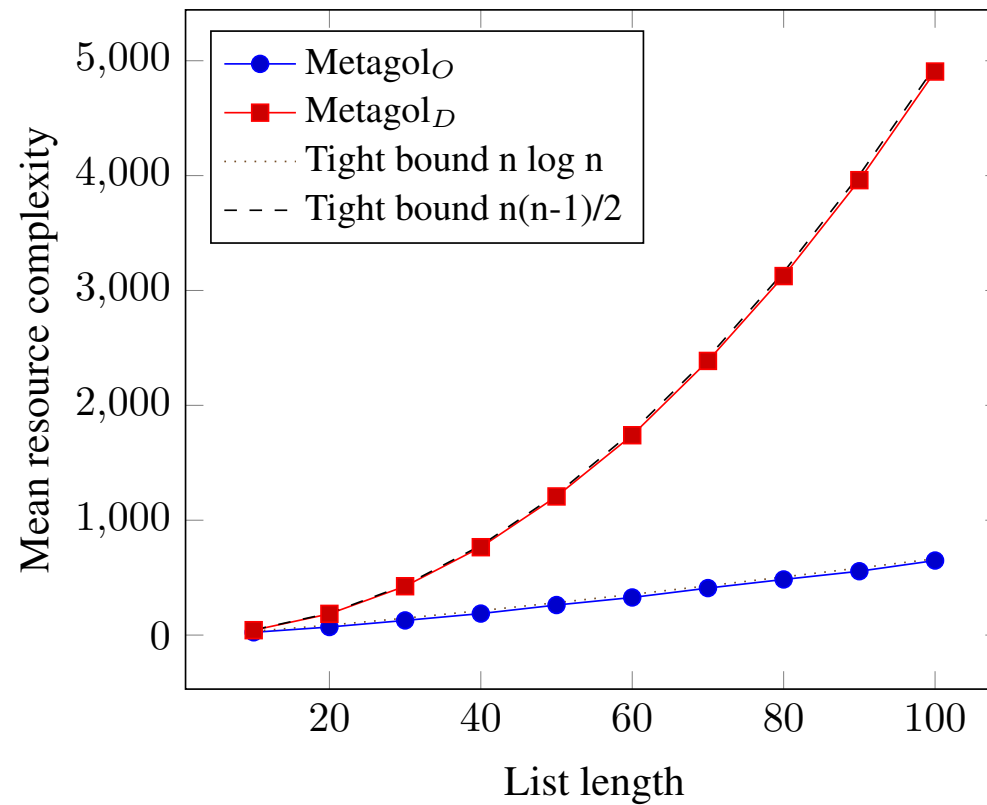
rs(A,B):- rs1(A,C), rs(C,B).

rs1(A,B):- pick(A,C), split(C,B).

rs1(A,B):- cmb(A,C), gst(C,B).

rs(A,B):- split(A,C), cmb(C,B).

Robot Letter Sorting mean resource complexity [Paper5.1]



Duplicate Character [Paper5.2]

Examples

`f([p,r,o,g,r,a,m],r).`

`f([i,n,d,u,c,t,i,o,n],i).`

Duplicate Character [Paper5.2]

Metagol_D - $O(n^2)$

`f(A,B):-head(A,B),f_1(A,B).`

`f(A,B):-tail(A,C),f(C,B).`

`f_1(A,B):-tail(A,C),element(C,B).`

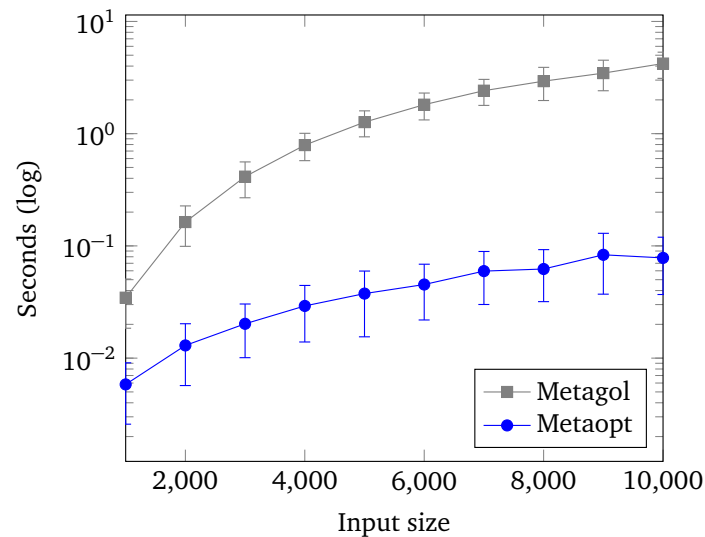
Metaopt - $O(n \log(n))$

`f(A,B):-msort(A,C),f_1(C,B).`

`f_1(A,B):-head(A,B),f_2(A,B).`

`f_1(A,B):-tail(A,C),f_1(C,B).`

`f_2(A,B):-tail(A,C),head(C,B).`



Framework - Cost function Φ [Paper5.2]

Metagol _O	$\sum_{e \in E} r(H, e)$
Metaopt	$\sum_{e \in E} \text{treecost}(H, e)$
General ordering	$\prec_{\Phi}$

Framework - Cost minimisation over Version Space [Paper5.2]

Dfn6	Version space $\mathcal{V}_{B,E}$	Hypothesis space consistent with B, E
Dfn7	Cost minimisation	$H \in \mathcal{V}_{B,E}$ and $\forall H' \in \mathcal{V}_{B,E} H \preceq_{\Phi} H'$

Metagol_O and Metaopt algorithm

Cost Minimisation

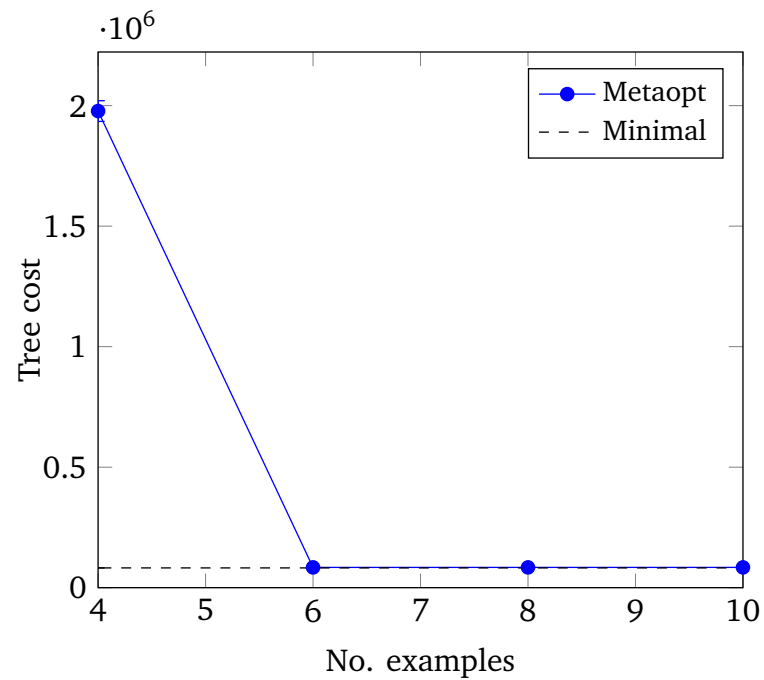
Iteration	Hypothesis
1	$ H_1 $ minimal in $\mathcal{V}_{B,E}$
$i > 1$	$ H_i $ minimal and $H_i \prec_{\Phi} H_{i-1}$
$i = \text{final}$	$\nexists H_i \ H_i \prec_{\Phi} H_{i-1}$
Return	$H_{\text{final}-1}$

Convergence theorem [Thm 1 Paper5.2]

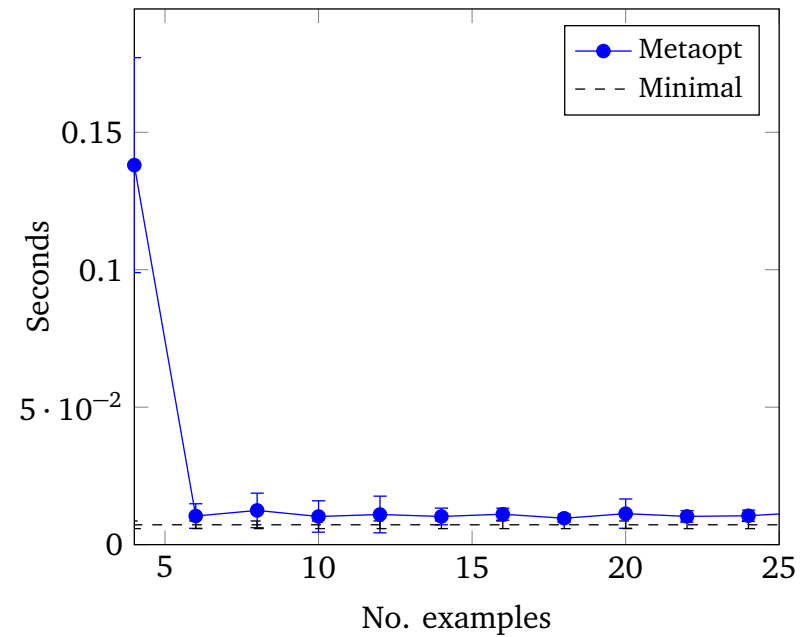
Given sufficiently large $|E|$

Metaopt returns $\inf_{\preceq_{\Phi}} \mathcal{V}_{B,E}$

Duplicate Character - Median Tree Costs [Paper5.2]

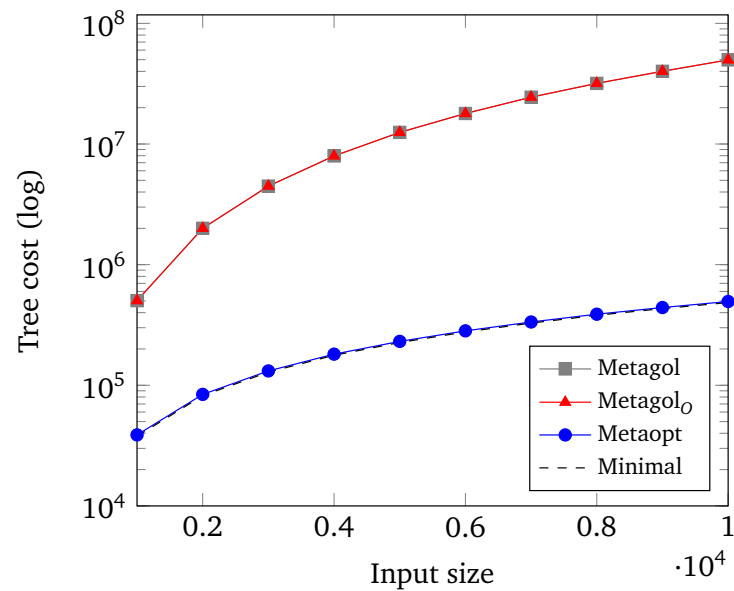


(a) Tree costs

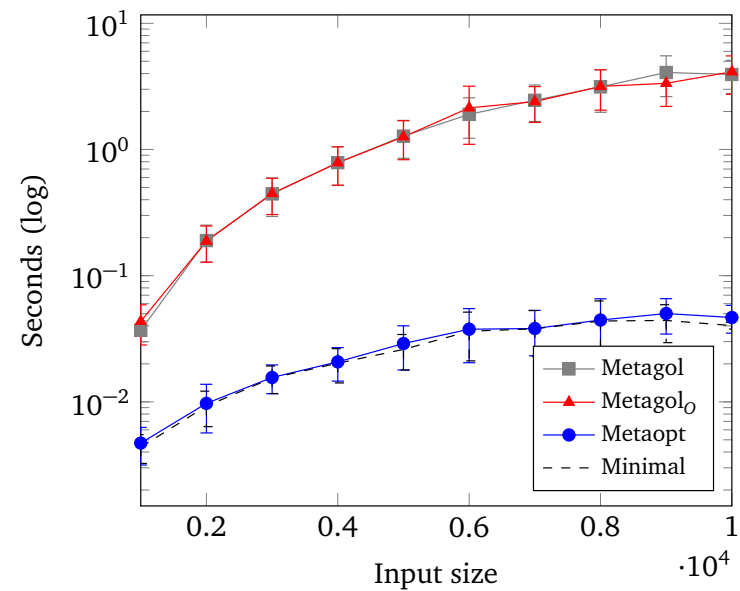


(b) Program runtimes

Duplicate Letter - Input size vs Tree Cost [Paper5.2]

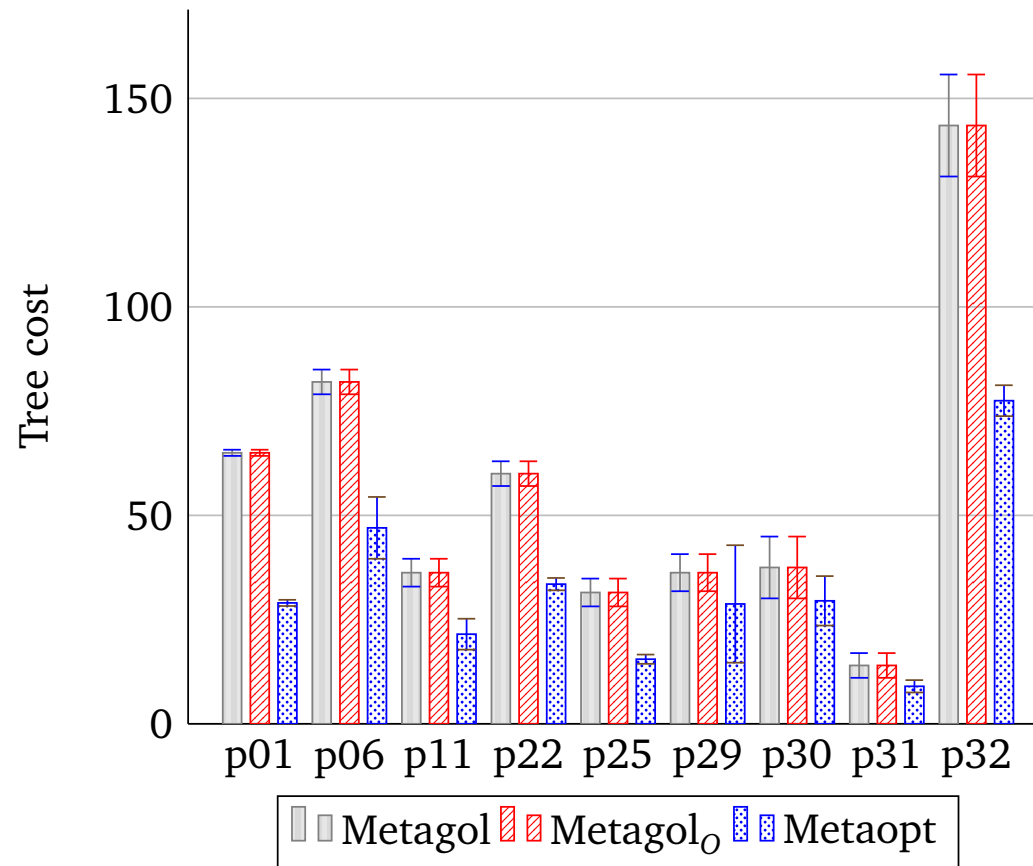


(a) Tree costs



(b) Program runtimes

String Transformations - Median Tree Costs [Paper5.2]



Summary

- Shorter programs fewer examples - Blumer bound
- Shorter programs not always most efficient
- Metagol₀ minimises robot energy cost
- Metaopt minimises SLD resolutions
- Both find minimal size program first
- Iteratively relax size minimising cost Φ
- Convergence theorem
- Efficiency - Postman, Sorting, Duplicate, String Transform