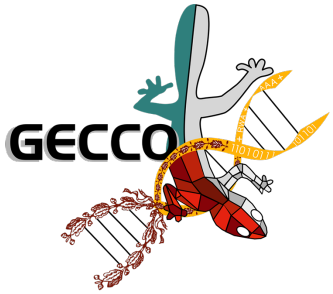


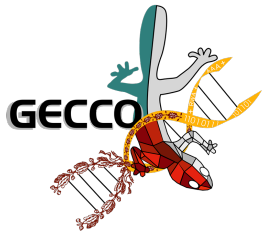
Transfer Learning in Evolutionary Spaces

Nelishia Pillay
Department of Computer Science
University of Pretoria
South Africa

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

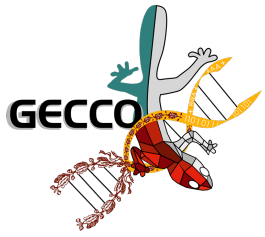
Tutorial Website

https://www.cs.up.ac.za/cs/npillay/TLEA.htm



Overview

- Evolutionary spaces
- Transfer learning
- Transfer learning in search
- Benefits of TL in search
- Case study: solution space
- Case study: program space
- Case study: heuristic space
- Case study: design space
- Automated TL in EAs
- Discussion: future research directions



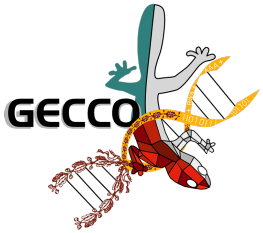
Evolutionary spaces

Solution space

Program space

Heuristic Space

Design Space

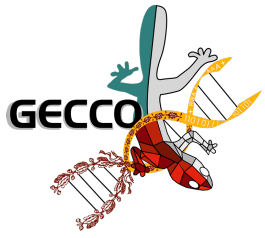


Transfer Learning

- Transfer of knowledge
- What to transfer?
- How to transfer?
- When to transfer?
- Positive vs. negative transfer
- Focus on data and feature transfer
- TL in EC

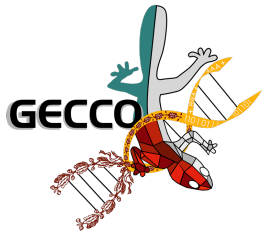
Transfer Learning in Search

- Knowledge learnt during search is transferred
- What is transferred?
  - Points in the search space
  - Elements of the population
  - Components of elements
  - Areas of the search space
- How is it transferred?
  - Forms part of the initial population of the target EA
- When is it transferred?
  - Last generations
  - Generation intervals



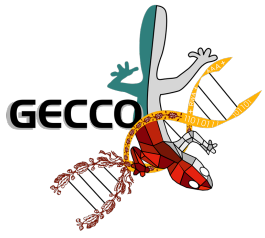
Benefits of TL in Search

- Improvement in performance
- Reduction in computational cost
- Reduction in the amount of data needed
- Improved convergence



Case Study: Solution Space

- Genetic algorithms for solving the travelling salesman problem (TSP)
- Symmetric TSP – Involves finding a route of minimum length that visits all cities exactly once and begins and ends at the same city. The distance between cities *m* and *n* is the same as the distance between *n* and *m*.
- Asymmetric TSP – Involves finding a route of minimum length that visits all cities exactly once and begins and ends at the same city. The distance between cities *m* and *n* is not necessarily the same as the distance between *n* and *m*.



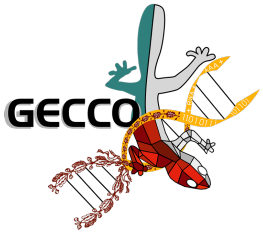
Case Study: Solution Space

- What is the aim of the transfer learning?
- Define the source and target domains
- What will be transferred?
- How will it be transferred?
- When will it be transferred?
- Transformation function needed?

Solution Space TSP
TL Case Study Discussion

Case Study: Program + Heuristic Space – Scheepers and Pillay (2021)

- Genetic programming generation construction hyper-heuristic
- Each point/element – a parse tree representing a heuristic
- Application one dimensional bin packing (1BPP)
- Aim – reduction in computational cost of solving more challenging problems
- Source domain – easy and medium 1 BPP problems
- Target domain – hard 1BPP problems



Case Study: Program + Heuristic Space – Scheepers and Pillay (2021)

- What to transfer
  - Population of last generation (TL1)
  - Best individuals of each generation (TL2)
  - Areas of the search space
    - Frozen root (TL 2.1)
    - Frozen second level (TL2.2)
    - Frozen leaf nodes (TL2.3)
- How to transfer?
- When to transfer?

Case Study: Program + Heuristic Space – Scheepers and Pillay (2021)

- Performance evaluation
  - Objective value – Number of bins
  - Computational effort – Koza computational effort formula

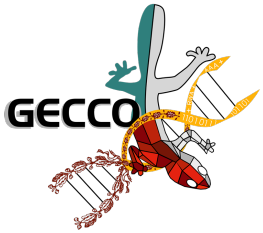
$$f(x, M, i) = R(x, M, i) * M * i$$

$$R(x, M, i) = \lceil \frac{log(1 - x)}{log(1 - P(M, i))} \rceil$$

  - Generality – Standard deviation of differences (SDD)

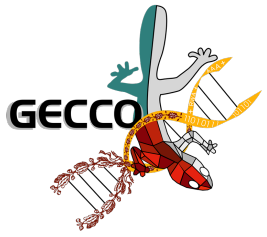
$$SDD(H) = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N - 1}}$$

where N is the number of problem instances, $x_i = 0$ if $o_i = 0$ and $b_i = 0$. Otherwise $x_i = (|o_i - b_i|)/average * 100$.



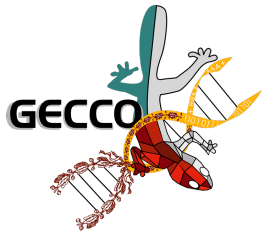
Case Study: Program + Heuristic Space – Scheepers and Pillay (2021)

- Objective value performance
  - TL approaches performed better
  - TL2, TL2.2 and TL2.3 produced the best results
  - Best approach of the three is problem dependent
- Computational effort performance
  - TL2 best computational effort
  - TL2.1, TL2.2, TL2.3 better than TL1
- Generality performance
  - TL1 and TL2 best generality
  - TL2.1, TL2.2, TL2.3 worst generality



Case Study: Heuristic Space

- Selection construction/perturbative genetic algorithm hyper-heuristic for educational timetabling
- Timetabling problems involve allocation of an entities, e.g. exams, classes to timetable slots to reduce had and soft constraints
- Educational timetabling
  - Examination timetabling
  - University course timetabling
  - School timetabling



Case Study: Heuristic Space

- What is the aim of the transfer learning?
- Define the source and target domains
- What will be transferred?
- How will it be transferred?
- When will it be transferred?

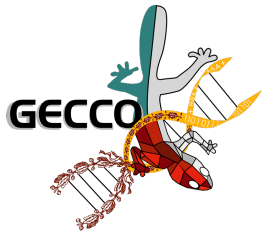
Heuristic Space Educational Timetabling
TL Case Study Discussion

Case Study: Heuristic Space (Singh and Pillay 2022)

- Ant colony optimization generation construction hyper-heuristic
- Aim: Reduction in computational cost
- Applications
  - Movie scene scheduling problem
  - Quadratic assignment problem
  - One dimensional bin packing
- Source domain: Simpler problem instance/s
- Target domain: Complicated problem instance/s

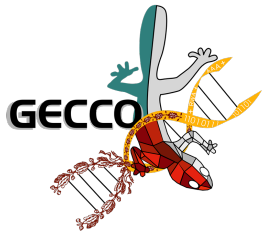
Case Study: Heuristic Space (Singh and Pillay 2022)

- What is transferred?
  - Pheromone maps from of the last iteration of the ACO
- How is it transferred?
  - Best pheromone map of the last iteration of the source ACO hyper-heuristic is transferred
- When is it transferred?
  - Used at the beginning of the target ACO hyper-heuristic instead of creating the maps randomly
- Performance
  - Drastic reduction in computational cost
  - Improvement in objective value for MSSP and QAP, poorer objective values for 1BPP



Case Study: Program Space

- Genetic programming for evolving prediction models for disease diagnosis
- Given relevant patient attributes the model predicts whether the patient has the disease or not
- Disease diagnosis
  - Heat disease
  - Diabetes
  - COVID-19



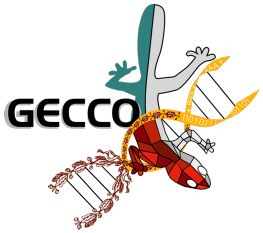
Case Study: Program Space

- What is the aim of the transfer learning?
- Define the source and target domains
- What will be transferred?
- How will it be transferred?
- When will it be transferred?

Program Space Prediction
TL Case Study Discussion

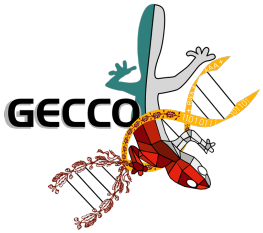
Case Study: Design Space – Nyathi and Pillay (2021)

- Automated design of the genetic programming algorithm to produce classifiers
- Design decisions represented in chromosome
  - Representation
  - Parameter values
  - Genetic operators
  - Control flow
- Grammatical evolution used for automated design (AutoGE)
- Source domain: NSL-KDD
- Target domain: UCI benchmark set



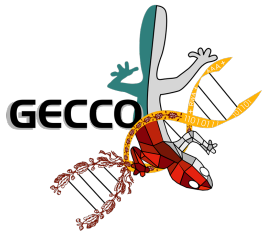
Case Study: Design Space – Nyathi and Pillay (2021)

- What is transferred?
  - Design of the GP classification algorithm
- How is it transferred?
  - Best performing designs from the source AutoGE to the target AutoGE
- When is it transferred?
  - Best performing designs form the initial population generation of the AutoGE
- Performance
  - Improved accuracy with using transfer learning
  - Reduction in computational cost



Differences/Similarities for Different Spaces

- Benefit/aim of transfer learning
- What to transfer?
- How to transfer?
- When to transfer?
- Performance
- Challenges



Automated Transfer Learning

- Automating TL design decisions
- Design decision
- Approaches
  - Selection perturbative hyper-heuristic – single point hyper-heuristic applied to randomly created design
  - Genetic algorithm – Each chromosome is a design
- Automated transfer learning for evolutionary algorithms (ATLEA)

ATLEA: Automated Transfer Learning in Evolutionary Algorithms


GECCO

ATLEA

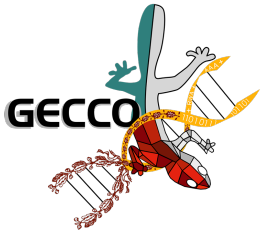

Source Evolutionary Algorithm
ATLEA
Target Evolutionary Algorithm

Automated Transfer Learning in ATLEA

- Selection perturbative hyper-heuristics
  - Random heuristic select, accept improving moves
  - Choice function, AILTA
- Genetic algorithm
- Extensible

ATLEA Design

- How many elements to transfer from each generation?
- Percentage of the element to transfer?
- Design syntax
  - A component for each generation
  - Number of elements to transfer
  - Percentage of the element to transfer



Basic Selection Perturbative Hyper-Heuristic

- Randomly select heuristic
- Accept if there is an improvement in performance
- Perturbative heuristics
  - swap
  - swapn
  - replace
- Abstract methods
  - evaluate
  - isBetter

Discussion: Future Research Directions