SBST for AI and beyond

from single inputs to generative models!

from here and into the future!

from the academic lab to the real (industrial) world!

SBST 2020 Live Keynote, 2020-07-02

Robert Feldt, robert.feldt@chalmers.se
Chalmers Univ of Tech, Gothenburg, Sweden

CHALMERS
UNIVERSITY OF TECHNOLOGY

Thanks to my many collaborators and friends!

Shin Yoo & his many **great and talented students**!

The late **Simon Poulding** who was a real intellectual and personal friend!

My closest SBST colleagues **Felix Dobslaw, Francisco Gomes, Greg Gay, & Richard Torkar** at Chalmers/GU!

…and many more/others…

Some kind of outline of today…

1. Future SBST (based on some pet peeves of mine… :))

2. What does AI/ML bring to the table?

3. A Manifesto for “Industrial SBST”

Part 1:

Let’s start from the pet peeves :)

PP1: We (still) use mainly (a few) EvoAlgs

Broadening the Search in Search-Based Software Testing: It Need Not Be Evolutionary

Robert Feldt and Simon Poulding
Dept. of Software Engineering
Belkinge Institute of Technology, Karlskrona, Sweden
Email: robert.feldt@bth.se and simon.poulding@bth.se

***Abstract*—Search-based software testing (SBST) can potentially help software practitioners create better test suites using less time and resources by employing powerful methods for search and optimization. However, research on SBST has typically focused on only a few search approaches and basic techniques. A majority of publications in recent years use some form of evolutionary search, typically a genetic algorithm, or, alternatively, some other optimization algorithm inspired from nature. This paper argues that SBST researchers and practitioners should not restrict themselves to a limited choice of search algorithms or approaches to optimization. To support our argument we empirically investigate three alternatives and compare them to the de facto SBST standards in regards to performance, resource efficiency and robustness on different test data generation problems: classic algorithms from the optimization literature, bayesian optimization with gaussian processes from machine learning, and nested monte carlo search from game playing / reinforcement learning. In all cases we show comparable and sometimes better performance than the current state-of-the-SBST-art. We conclude that SBST researchers should consider a more general set of solution approaches, more consider combinations and hybrid solutions and look to other areas for how to develop the field.**

I. Introduction

published at these venues in 2013 and 2014. Two-thirds of the papers—26 out of 39—applied an evolutionary algorithm, of which 23 applied a Genetic Algorithm (GA): 15 as a standard GA, 2 as a GA-based memetic algorithm, and 7 as a multi-objective GA (the majority using NSGA-II)[1]. The next most-frequently applied algorithms were Genetic Programming (4 papers), (1+1) EA (4 papers), hill-climbing (3 papers), and alternating variable methods (3 papers). Our analysis suggests that evolutionary search, and GAs in particular, are the algorithms of choice for both single- and multi-objective problems in SBST.

We offer a number of explanations for this prevalence of GAs as the search technique. GAs can be applied to a wide range of problem classes and typically find solutions with acceptably good quality. This wide applicability permits us, as researchers, to re-use the knowledge gained in applying GAs to one testing problem when solving subsequent problems. In addition, there is a great deal of active research in GAs that can guide their application to testing problems, and this research is typically disseminated in a form that is readily-accessible to us. In contrast, the research on classic optimization algorithms

2015: EA's and GA's are used in 60-80% of papers

We checked 39 SBST papers from:

- SBST 2013 & 2014
- GECCO SBSE 2013 & 2014
- SSBSE 2013 & 2014

2015: Flexibility of EvoAlgs not always called for!

TABLE II
OPTIMIZATION ALGORITHMS USED AND THEIR MAIN CHARACTERISTICS

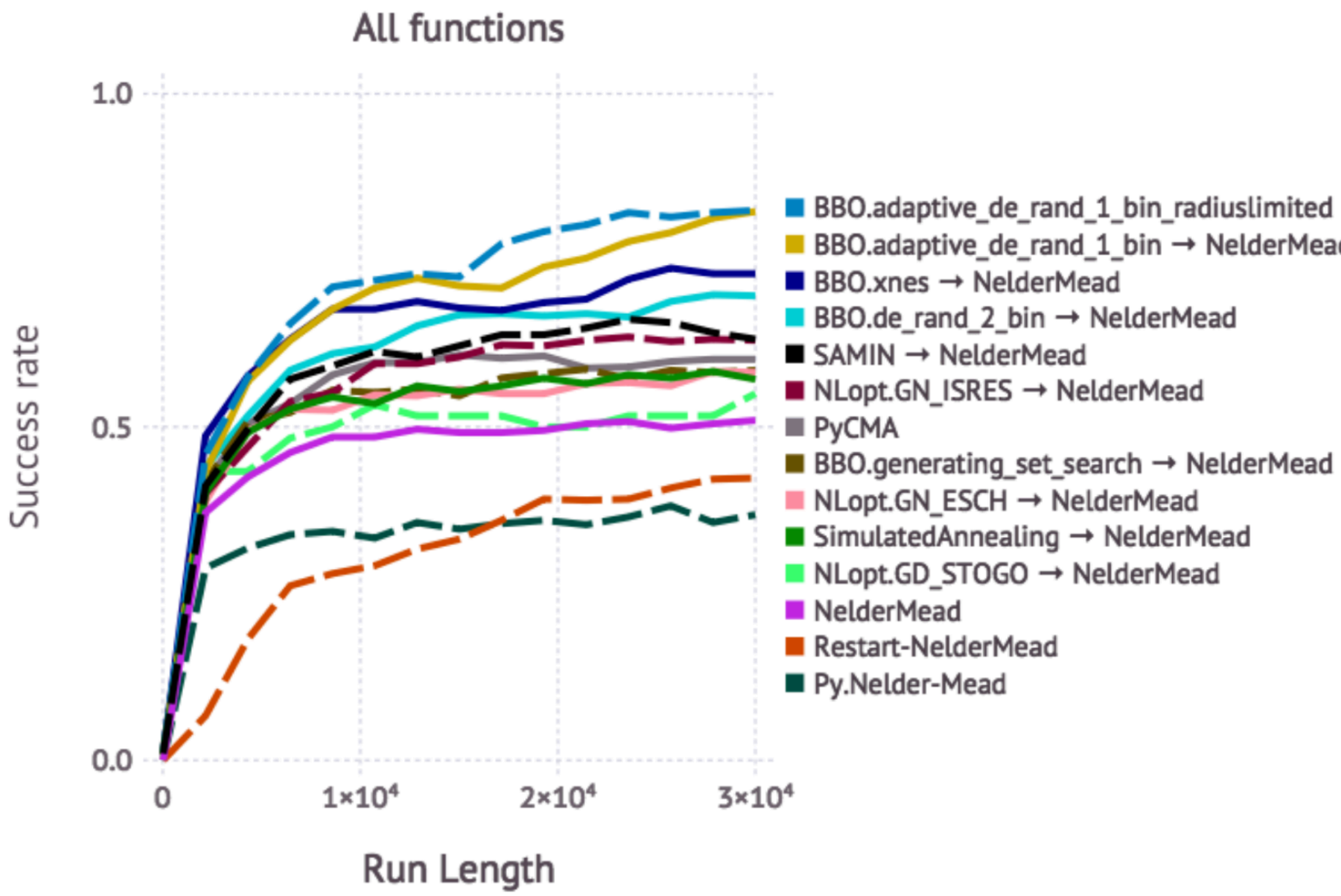
Name	**Description**	**Refs**	**Type**	**Library**
DE	Differential evolution (rand/1/bin)	[13]	Evo	BlackBoxOptim
SRES+BOB	Stochastic Ranking Evo Strategy, then BOBYQA	[14], [15]	Evo. Combination	NLopt
ESCH+COB	Evo Strategy, then COBYLA	[16], [17]	Evo. Combination	NLopt
CRS+BOB	Controlled Random Search (CRS), then BOBYQA	[18], [19], [15]	Combination	NLopt
Compass	Direct (compass) search with adaptive step size	[11]	Non-Evo	BlackBoxOptim
RandSrch	Random Search (baseline)	N/A	Non-Evo	BlackBoxOptim

Algorithm	**Rank**	**Wins**	**MAPE**	**HitRate**	**Time**
ESCH+COB	**2.2**	**6**	**29.7**	5.5	**705.3**
CRS+BOB	3.21	4	44.5	**8.5**	1203.7
DE	3.24	2	43.6	4.0	985.9
RandSrch	3.8	0	46.9	1.8	1145.3
Compass	3.9	0	49.3	2.1	1168.6
SRES+BOB	4.9	0	64.3	3.5	1583.1
Unoptimized	6.7	0	164.79	0.2	0.0

Population-based EvoAlg + Traditional “local” tuning!

Benchmark results

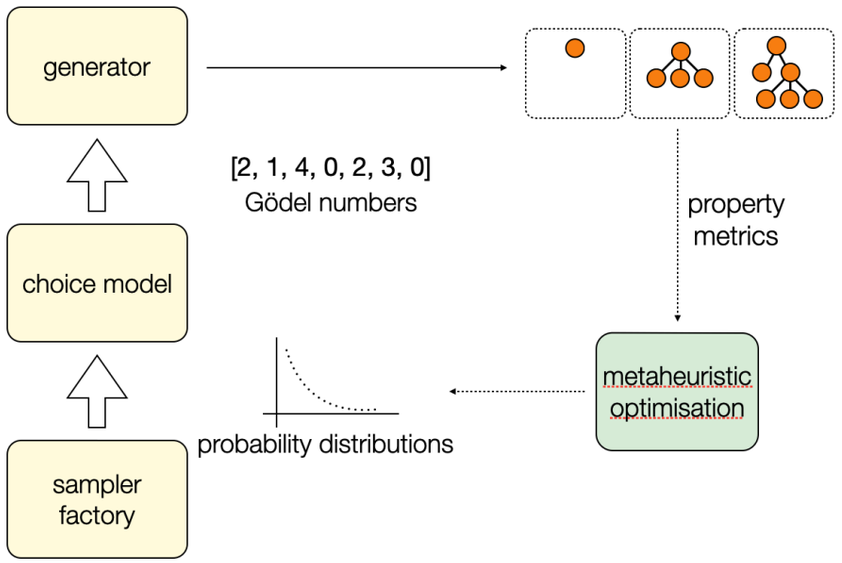
The average sucess rate (meaning the optimizer reached the minimum + 1e-6) in function of the number of objective function evaluations :



https://github.com/jonathanBieler/BlackBoxOptimizationBenchmarking.jl

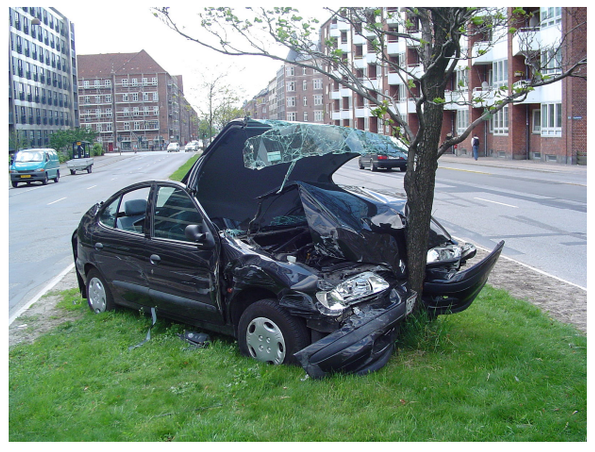
PP2: We often search only for single inputs

Models:

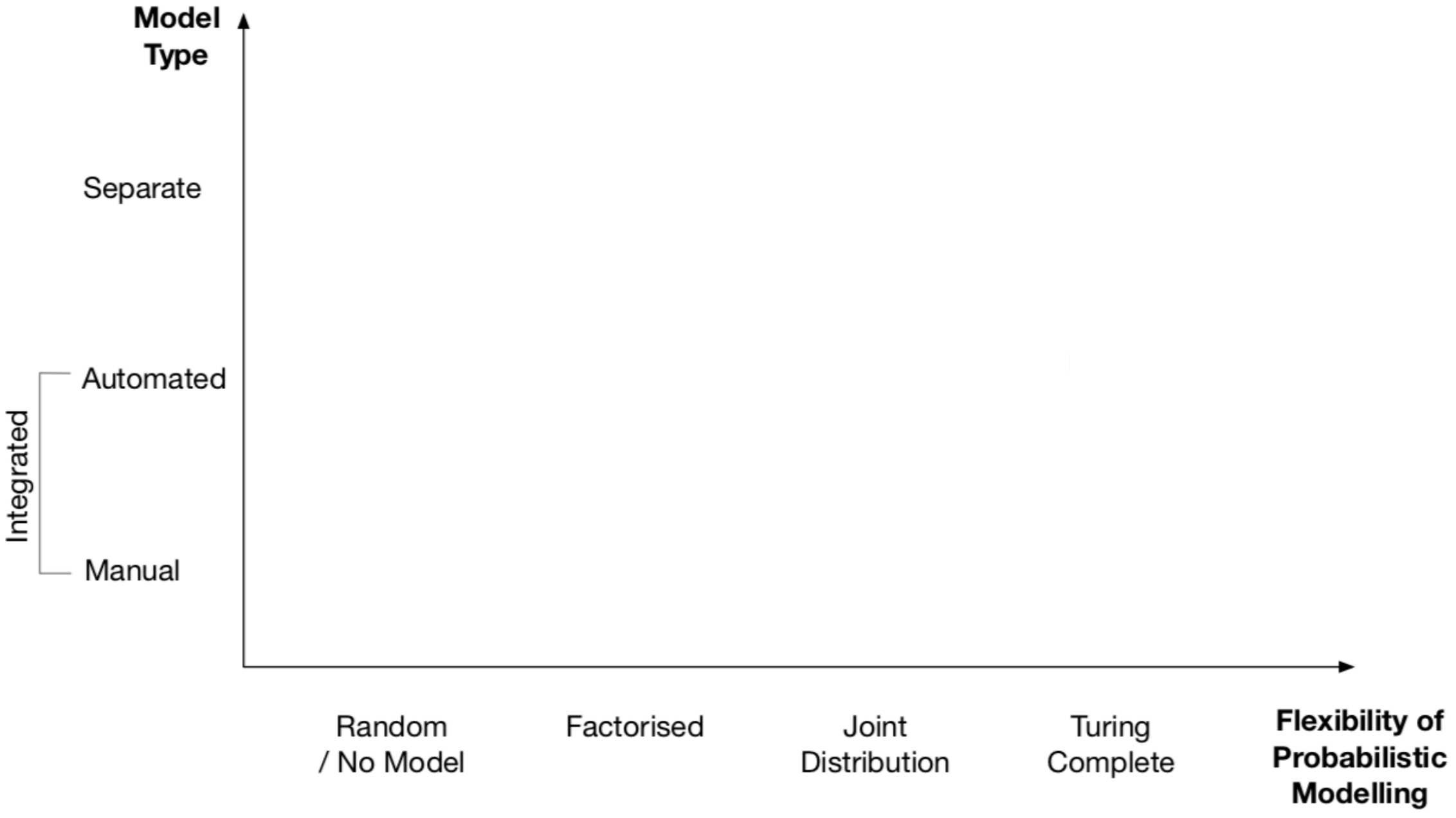


Sets: *map(SUT,*  *)*

Individual input: *SUT(*  *)* =>



PP2: Search for higher-level models instead!



PP3: Minor variations instead of fundamentals

ELSEVIER

Advances in Engineering Software

Volume 69, March 2014, Pages 46-61

Grey Wolf Optimizer

EC
29,5

464

Received 28 April 2011
Revised 5 August 2011
Accepted 11 August 2011

Bat algorithm: a novel approach for global engineering optimization

Springer Link

Original Article | Published: 17 March 2015

Multi-Verse Optimizer: a nature-inspired algorithm for global optimization

PP3: Minor variations instead of fundamentals

https://en.wikipedia.org/wiki/List_of_metaphor-based_metaheuristics

1 Algorithms
- 1.1 Simulated annealing (Kirkpatrick et al. 1983)
- 1.2 Ant colony optimization (Dorigo, 1992)
- 1.3 Particle swarm optimization (Kennedy & Eberhart 1995)
- 1.4 Harmony search (Geem, Kim & Loganathan 2001)
- 1.5 Artificial bee colony algorithm (Karaboga 2005)
- 1.6 Bees algorithm (Pham 2005)
- 1.7 Glowworm swarm optimization (Krishnanand & Ghose 2005)
- 1.8 Shuffled frog leaping algorithm (Eusuff, Lansey & Pasha 2006)
- 1.9 Cat Swarm Optimization (Chu, Tsai, and Pan 2006)
- 1.10 Imperialist competitive algorithm (Atashpaz-Gargari & Lucas 2007)
- 1.11 River formation dynamics (Rabanal, Rodríguez & Rubio 2007)
- 1.12 Intelligent water drops algorithm (Shah-Hosseini 2007)
- 1.13 Gravitational search algorithm (Rashedi, Nezamabadi-pour & Saryazdi 2009)
- 1.14 Cuckoo search (Yang & Deb 2009)
- 1.15 Bat algorithm (Yang 2010)
- 1.16 Spiral optimization (SPO) algorithm (Tamura & Yasuda 2011,2016-2017)
- 1.17 Flower pollination algorithm (Yang 2012)
- 1.18 Cuttlefish optimization algorithm (Eesa, Mohsin, Brifcani & Orman 2013)
- 1.19 Heterogeneous Distributed Bees Algorithm (Tkach et al., 2013)
- 1.20 Cooperative Group Optimization (2014)
- 1.21 Artificial swarm intelligence (Rosenberg 2014)
- 1.22 Colliding bodies optimization (Kaveh and Mahdavi 2014)
- 1.23 Duelist Algorithm (Biyanto 2016)
- 1.24 Harris hawks optimization (Heidari et al. 2019)
- 1.25 Killer Whale Algorithm (Biyanto 2016)
- 1.26 Rain Water Algorithm (Biyanto 2017)
- 1.27 Mass and Energy Balances Algorithm (Biyanto 2018)
- 1.28 Hydrological Cycle Algorithm (Wedyan et al. 2017)
- 1.29 Emperor Penguins Colony (Harifi et al. 2019)
- 1.30 Momentum Balance Algorithm (MBA) (Biyanto et al. 2019)
- 1.31 Shuffled Shepherd Otimization Algorithm (SSOA) (Kaveh and Zaerreza 2020)
- 1.32 A mayfly optimization algorithm (MA) (Zervoudakis & Tsafarakis 2020)

PP3: Minor variations instead of fundamentals

Criticism of the metaphor methodology [edit]

In response, Springer's *Journal of Heuristics* has updated their editorial policy to state that:[81]

> Proposing new paradigms is only acceptable if they contain innovative basic ideas, such as those that are embedded in classical frameworks like genetic algorithms, tabu search, and simulated annealing. The Journal of Heuristics avoids the publication of articles that repackage and embed old ideas in methods that are claimed to be based on metaphors of natural or manmade systems and processes. These so-called "novel" methods employ analogies that range from intelligent water drops, musicians playing jazz, imperialist societies, leapfrogs, kangaroos, all types of swarms and insects and even mine blast processes (Sörensen, 2013). If a researcher uses a metaphor to stimulate his or her own ideas about a new method, the method must nevertheless be translated into metaphor-free language, so that the strategies employed can be clearly understood, and their novelty is made clearly visible. (See items 2 and 3 below.) Metaphors are cheap and easy to come by. Their use to "window dress" a method is not acceptable."
>
> [...] Implementations should be explained by employing standard optimization terminology, where a solution is called a "solution" and not something else related to some obscure metaphor (e.g., harmony, flies, bats, countries, etc.).
>
> [...] The Journal of Heuristics fully endorses Sörensen's view that metaphor-based "novel" methods should not be published if they cannot demonstrate a contribution to their field. Renaming existing concepts does not count as a contribution. Even though these methods are often called "novel", many present no new ideas, except for the occasional marginal variant of an already existing methodology. These methods should not take the journal space of truly innovative ideas and research. Since they do not use the standard optimization vocabulary, they are unnecessarily difficult to understand.

Maybe SBST/SBSE/AI4SE venues should have similar restrictions!?

Part 2:

What does AI/ML bring to the table?

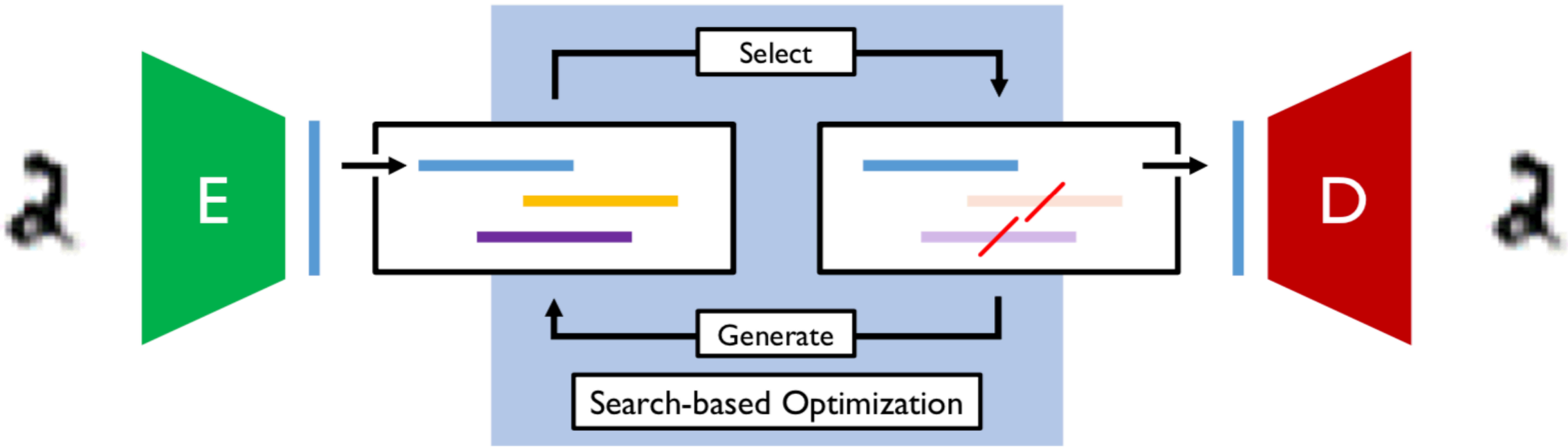
AI/ML very aligned with SBST!

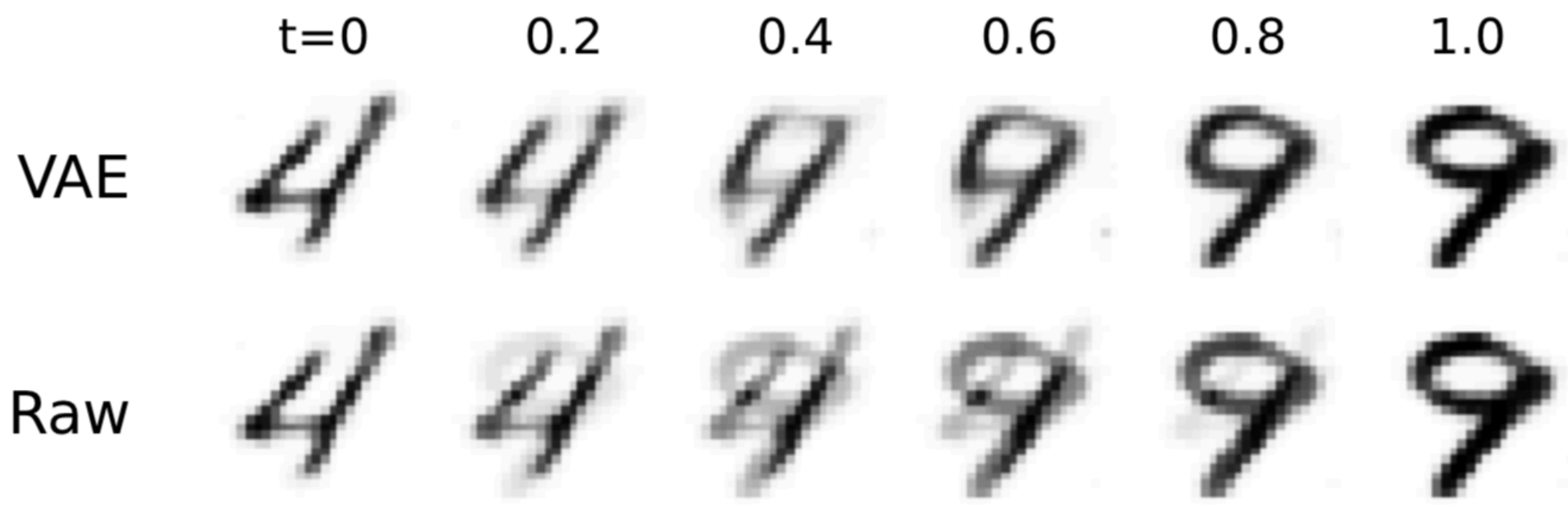
1. Fundamental connections and opportunities:
   - Soft/fuzzy objectives/requirements
   - Embarrassingly parallel
   - Few existing, good solutions for testing AI/ML

2. Many challenges though:
   - How avoid costly loop within costly loop?
   - More complex “inputs” than for traditional SW
   - Practical tools and not only papers
   - …

3. Key solutions & “Killer apps”:
   - Hybridize search “into” the AI/ML model workings
   - Complex generative models to set up “scenes” and simulations
   - Search4SE but also SE4Search!?

Example: SINVAD searches the latent space





Example: Finding high-risk simulation scenarios

Old funding application:

STRESS – Simulation-Based Testing of Smart Systems of Systems

Robert Feldt, Paul Pettersson, Daniel Sundmark,
Birgitta Lindstrom, and Christian Berger
with contributions by Jeff Offutt, Simon Poulding, and Wasif Afzal

Blekinge Tekniska Högskola, SE-371 79 Karlskrona, Sweden
robert.feldt@bth.se

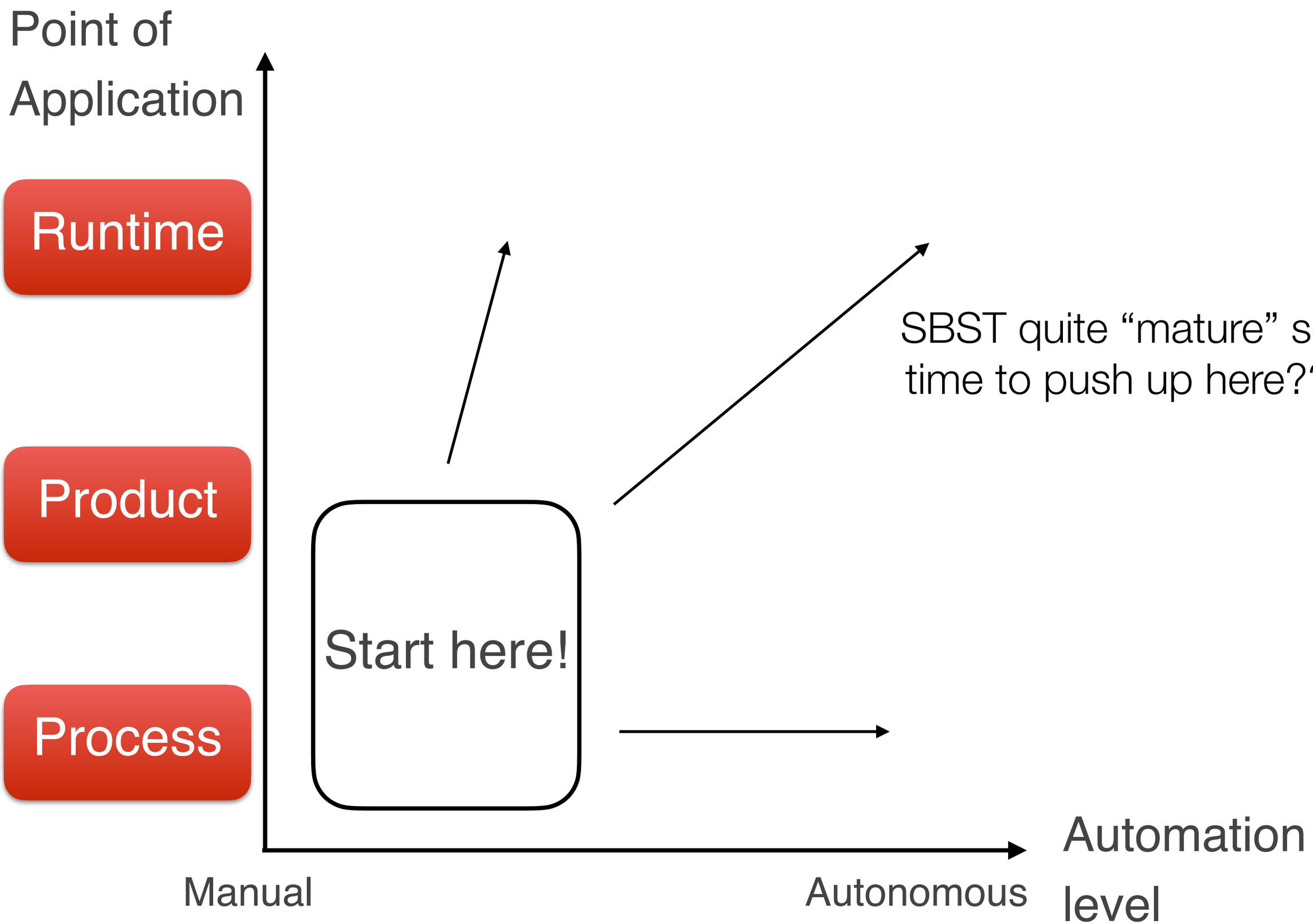
2.2.4 WP4: Identifying High-Risk Scenarios Using Metaheuristic Search [BTH (lead), SICS, MdH]

Research Challenge: The research challenge is to identify, through simulation, high-risk scenarios – configurations of the smart system, its environment, and the task it performs in that environment – that lead to undesirable behaviour such as a smart vehicle colliding with another vehicle or injuring its occupants. Not only does this identify potential problems with the system design and implementation earlier in the development process, but these scenarios, especially when combined with the root cause analysis of WP1, can identify a small set of tests that will be cost-effective during more expensive real-world testing.

Methodology: Search-based software testing (SBST) re-interprets the problem of generating test scenarios as an optimisation problem: a fitness metric is used to quantify how close a candidate scenario is to having desired properties, and sophisticated metaheuristic search algorithms are used to optimise this fitness metric. Research over the last fifteen years has shown SBST to be a high-effective and low-cost technique for generating test scenarios. A testing scenario for a smart vehicle will be highly complex yet constrained to be realistic by the model of the environment created in WP1 and the model for its perception interface as developed in WP2. For this reason, the work package will build on existing work by Feldt and Poulding on using SBST for generating highly complex, structured testing scenarios [4], and initial work on deriving scenarios for testing autonomous robot vehicles [9]. The work package will identify and evaluate fitness metrics and search algorithm that efficiently guide the search to high-risk scenarios and, using the demonstrators, evaluate and improve on the extent to which high-risk scenarios identified by search remain valid in the real-world. In addition to the participating in-project resources

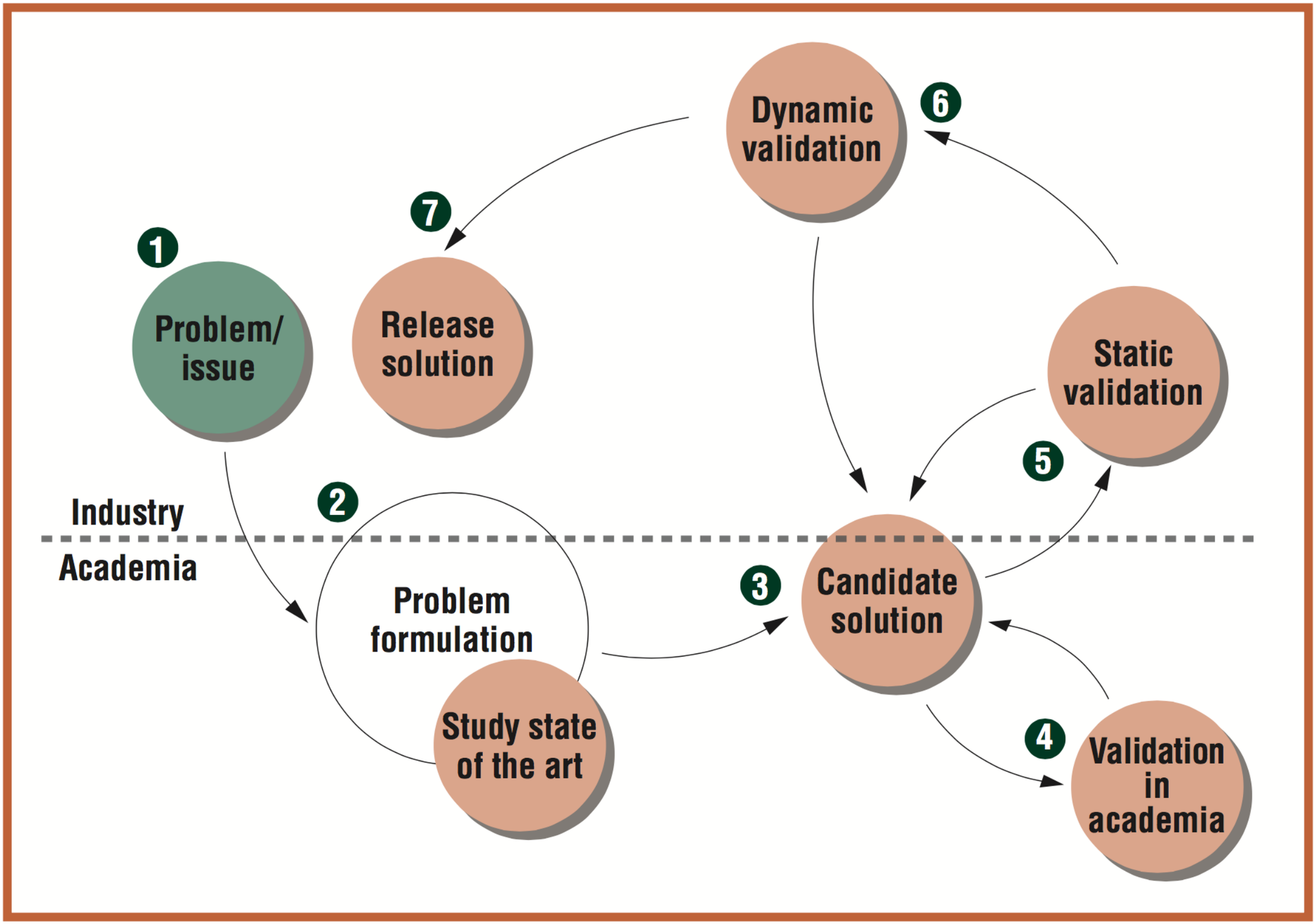
Think about: Relation AI <-> SE <-> Search

AI-in-SE applications have different levels of risk/gain

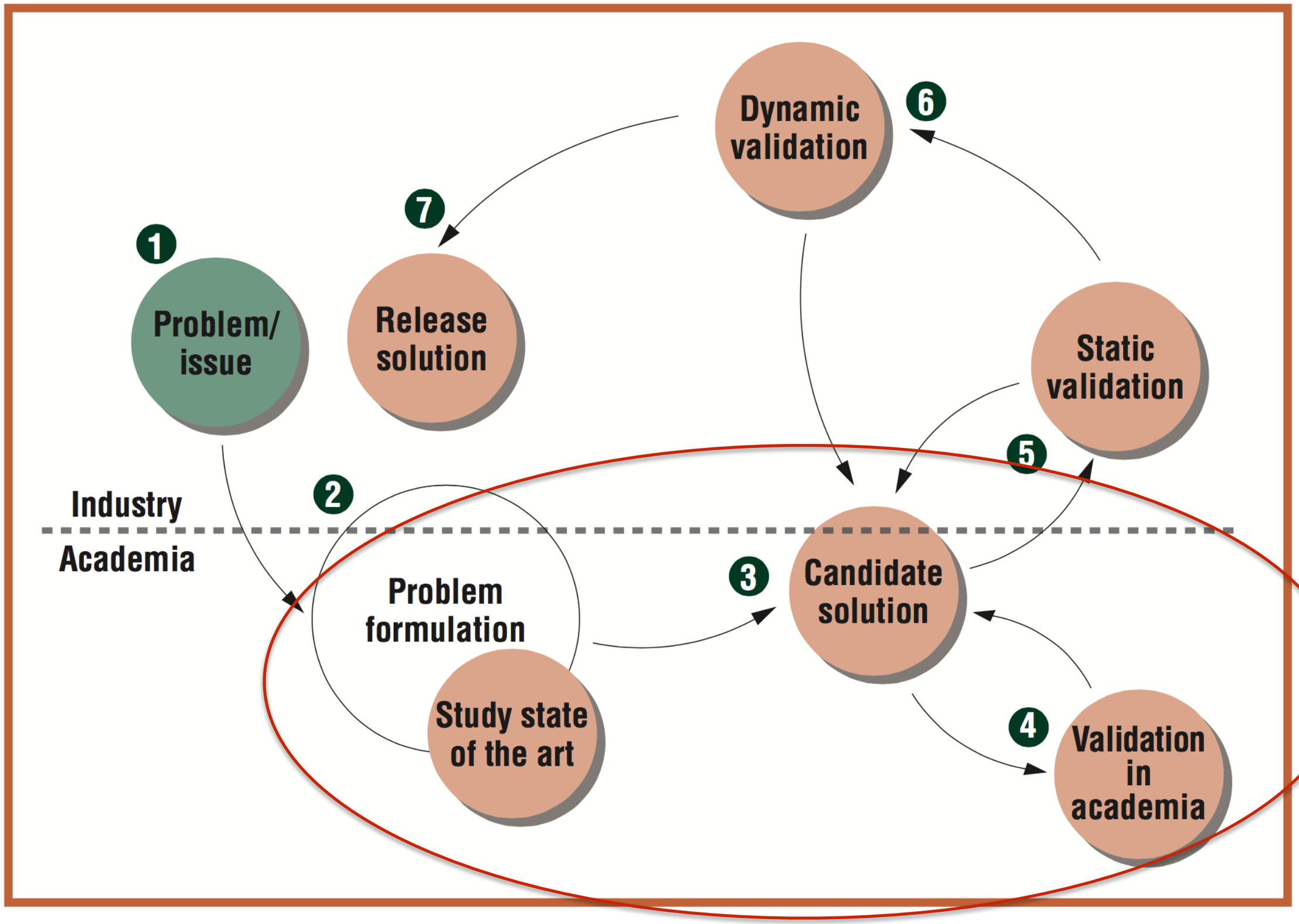


Part 3:
A Manifesto for “Industrial SBST”

Technology Transfer Model



SBST has stayed mostly within academia!



Manifesto for Industrial SBST 0.1

Through systematic research we are uncovering a science of search-based software testing so that we can better help software practitioners. Through this work we have come to value:

Augmenting humans over automating them away

Adaptive toolbox & hybridisation over one-alg-fits-all

Trusting & listening over data extraction & “preaching”

Patience over quick results or low-hanging fruits

Information focus over search/algorithm focus

Information costs over time over include-it-all and search

Multi-objective over single-objective

Interpretability & Scalability over accuracy & complexity

Information costs over time over include-it-all and search

1. Many SBST/SBSE/AI4SE solutions include:
   - multiple
   - relational
   - diverse info sources.

2. This leads to
   - high up-front costs
   - very high maintenance costs
   - requires synchronisation between many parts of org
   - requires high edu/training costs, and
   - makes interpretation of results more complex and costly.

Real-world example: One company I worked with tried dynamic approaches (instrumentation) then formally BANNED their use in the company!

Information costs over time over include-it-all and search

3. So we must be SUPER-sure:
   - that **each added information source is motivated**
   - and its **value/cost trade-off is known and clear**.

4. Proposal: **Information Source Ablation (ISA)**
   - empirically investigate **sequence of more complex models** (using increasingly more/complex information srcs)
   - and **show practitioners the trade-off** in value vs cost/complexity, and
   - **let them select** the best trade-off for their org.
   - *Researcher with no industry access: Do ISA anyway so practitioners can see it in your paper.*

Multi-objective over single-objective

1. If you think your problem can be meaningfully captured in a single objective, think again!

2. In industry nothing is ever that simple
   - There are always many conflicting objectives
   - 2-3 are rarely enough

3. So we just use NSGA-II, right?
   - Sure, maybe that is fine but 2-3 objectives are rarely enough
   - and NSGA-II is rather old, we should be more aware of S-O-T-A

4. Meaningfully handle the non-dominated result set!!
   - NOT: MO & then select one solution to compare to baselines!
   - Better: Define a few realistic use scenarios and consider parts of “Pareto front” that caters to each one
   - Best: Get feedback from practitioners

Multi-objective over single-objective

Real-world objectives in test selection / prioritization:

1. ChangedFunctionCoverage (Maximize)
2. SubsetSize (Minimize)
3. AggregatedFailRate (Maximize, predicted or historical)
4. TestFeatureCoverage (Maximize)
5. HardwareSetupCoverage (Maximize)
6. TestExecutionTime (Minimize, predicted or historical)
7. DaysSinceLastTestExecution (Maximize)
8. ChangedFunctionCallTimes (Maximize)

Summary

Use multiple different SOTA searchers and hybridise them!

Search for models (and models of models), not single datums!

Focus on fundamental questions, not minor (search) variations!

Hybridize search/opt into AI/ML!

Investigate flexible, probabilistic models for generating complex inputs!

Search4SE now old but what about SE4Search!?

Philosophy & attitude as important as technology/search!

First consider info sources and (multi) objectives!

Then consider representations and search algs!

Keep it simple & optimise value vs complexity/cost (w. feedback)!