FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Michal Pospěch

Thesis title is N/A

Department of Theoretical Computer Science and Mathematical Logic

Supervisor of the bachelor thesis:	Mgr. Roman Neruda, CSc.
Study programme:	Computer science
Study branch:	General computer science

Prague 2021

Dedication. It is nice to say thanks to supervisors, friends, family, book authors and food providers.

Title: Thesis title is N/A

Author: Michal Pospěch

Department: Department of Theoretical Computer Science and Mathematical Logic

Supervisor: Mgr. Roman Neruda, CSc., Department of Theoretical Computer Science and Mathematical Logic

Abstract: Abstracts are an abstract form of art. Use the most precise, shortest sentences that state what problem the thesis addresses, how it is approached, pinpoint the exact result achieved, and describe the applications and significance of the results. Highlight anything novel that was discovered or improved by the thesis. Maximum length is 200 words, but try to fit into 120. Abstracts are often used for deciding if a reviewer will be suitable for the thesis; a well-written abstract thus increases the probability of getting a reviewer who will like the thesis.

Keywords: key words

Contents

Introduction

Introduction should answer the following questions, ideally in this order:

1. What is the nature of the problem the thesis is addressing?
2. What is the common approach for solving that problem now?
3. How this thesis approaches the problem?
4. What are the results? Did something improve?
5. What can the reader expect in the individual chapters of the thesis?

Expected length of the introduction is between 1–4 pages. Longer introductions may require sub-sectioning with appropriate headings — use `\section*` to avoid numbering (with section names like ‘Motivation’ and ‘Related work’), but try to avoid lengthy discussion of anything specific. Any “real science” (definitions, theorems, methods, data) should go into other chapters.

You may notice that this paragraph briefly shows different “types” of ‘quotes’ in TeX, and the usage difference between a hyphen (-), en-dash (–) and em-dash (—).

It is very advisable to skim through a book about scientific English writing before starting the thesis. I can recommend ‘*Science research writing for non-native speakers of English*’ by Glasman-Deal [1].

Chapter 1

Important first chapter

First chapter usually builds the theoretical background necessary for readers to understand the rest of the thesis. You should summarize and reference a lot of existing literature and research.

You should use the standard *citations*.

Use \emph command like this, to highlight the first occurrence of an important word or term. Reader will notice it, and hopefully remember the importance.

Obtaining bibTeX citation Go to Google Scholar[1], find the relevant literature, click the tiny double-quote button below the link, and copy the bibTeX entry.

This footnote is an acceptable way to 'cite' webpages or URLs. Documents without proper titles, authors and publishers generally do not form citations. For this reason, avoid citations of wikipedia pages.

Saving the citation Insert the bibTeX entry to the file `refs.bib`. On the first line of the entry you should see the short reference name — from Scholar, it usually looks like `author2015title` — you will use that to refer to the citation.

Using the citation Use the `\cite` command to typeset the citation number correctly in the text; a long citation description will be automaticaly added to the bibliography at the end of the thesis. Always use a non-breakable space before the citing parenthesis to avoid unacceptable line breaks:

```
Trees utilize gravity to invade ye
noble sires~\cite{newton1666apple}.
```

Why should I bother with citations at all? For two main reasons:

- You do not have to explain everything in the thesis; instead you send the reader to refer to details in some other literature. Use citations to simplify the detailed explanations.

[1] `https://scholar.google.com`

- If you describe something that already exists without using a citation, the reviewer may think that you *claim* to have invented it. Expectably, he will demand academic correctness, and, from your perspective, being accused of plagiarism is not a good starting point for a successful defense. Use citations to give the credit to people who have invented what you build upon.

How many citations should I use? Cite any non-trivial building block or assumption that you use, if it is published in the literature. You do not have to cite trivia, such as the basic definitions taught in the introductory courses.

The rule of thumb is that you should read, understand and briefly review at least around 4 scientific papers. A thesis that contains less than 3 sound citations will spark doubt in reviewers.

There are several main commands for inserting citations, used as follows:

- Knuth [2] described a great system for typesetting theses.
- We are typesetting this thesis with LaTeX, which is based on TeX and METAFONT [2].
- TeX was expanded to LaTeX by Lamport [3], hence the name.
- Revered are the authors of these systems! [2, 3]

1.1 Some extra assorted hints before you start writing English

Strictly adhere to the English word order rules. The sentences follow a fixed structure with subject followed by a verb and an object (in this order). Exceptions to this rule must be handled specially, and usually separated by commas.

Mind the rules for placing commas:

- Use the *Oxford comma* before 'and' and 'or' at the end of a longer, comma-separated list of items. Certainly use it to disambiguate any possible mixtures of conjunctions: *'The car is available in red, red and green, and green versions.'*
- Do not use the comma before subordinate clauses that begin with 'that' (like this one). English does not use subordinate clauses as often as Slavic languages because the lack of a suitable word inflection method makes them hard to understand. In scientific English, try to avoid them as much

as possible. Ask doubtfully whether each 'which' and 'when' is necessary — most of these helper conjunctions can be removed by converting the clause to non-subordinate.

  As an usual example, *'The sentence, which I wrote, seemed ugly.'* is perfectly bad; slightly improved by *'The sentence that I wrote seemed ugly.'*, which can be easily reduced to *'The sentence I wrote seemed ugly.'*. A final version with added storytelling value could say *'I wrote a sentence but it seemed ugly.'*

- Consider placing extra commas around any parts of the sentence that break the usual word order, especially if they are longer than a single word.

Do not write long sentences. One sentence should contain exactly one fact. Multiple facts should be grouped in a paragraph to communicate one coherent idea. Paragraphs are grouped in labeled sections for a sole purpose of making the navigation in the thesis easier. Do not use the headings as 'names for paragraphs' — the text should make perfect sense even if all headings are removed. If a section of your text contains one paragraph per heading, you might have wanted to write an explicit list instead.

Every noun needs a determiner ('a', 'the', 'my', 'some', …); the exceptions to this rule, such as non-adjectivized names and indeterminate plural, are relatively scarce. Without a determiner, a noun can be easily mistaken for something completely different, such as an adjective or a verb.

Consult the books by Glasman-Deal [1] and Sparling [4] for more useful details.

Chapter 2

Theory

2.1 Reinforcement learning

- problem description (state space, action space, reward...)
- various methods
  - value function
  - criterion of optimality
  - direct policy search (and various methods)

2.2 Evolutionary algorithms

Evolutionary algorithms (EA) are a type of optimisation metaheuristics inspired by the process of bilogical evolution. At first a number of possible solutions to the problem at hand is generated (*population*) and each solution (*individual*) is encoded (via a domain-specific encoding) and evaluated giving us the value of its *fitness*. Fitness is a function describing how good that particular individual is and it is everything that is needed for creation of Then a new population is created using a *crossover* (combination) of 1 or more individuals which are selected using the operator of *parental selection*. Each of the newly created individuals has a chance to be mutated via the *mutation* operator. Finally a new population is selected from *offsprings* and possibly the parents based of fitness and enters the next iteration of the EA and the following generation is chosen using *environmental selection* operator. The algorithm repeats until the stop condition is met, usually a set number of iterations or small improvement of fitness between 2 generations.

There are many variands of EAs such as genetic algorithms (most common), genetic programming, evolutionary programming, neuroevolution or evolutionary strategies that are further described in following chapter. [5] [6]

Algorithm 1 Evolutionary algorithm

```
1: initialize population $P^0$ with $n$ individuals
2: set $t = 0$
3: repeat
4:     $Q^t = \{\}$
5:     for $i \in \{1 \dots m\}$ do
6:         $p_1, \dots, p_\rho = ParentalSelection(P^t)$
7:         $q = Crossover(p_1, \dots, p_\rho)$
8:         $q = Mutation(q)$ with chance $p$
9:         $Q^t = q \cup Q^t$
10:    end for
11:    $P^{t+1} = EnvironmentalSelection(Q^t \cup P^t)$
12:    increment $t$
13: until stop criterion fullfilled
```

2.3 Evolutionary strategies

Evolutionary strategies (ES) are a type of optimisation metaheuristic which further specialises EA and restricts their level of freedom. The selection for crossover is unbiased, mutation is parametrised and thus controllable, individuals which should be put to next generation are chosen ordinally based on fitness and individuals contain not only the problem solution but also control parameters.

More formally ES $(\mu/\rho, \kappa, \lambda)$ has μ individuals in each generation, which produces λ offsprings, each created by crossover of ρ individuals and each individual is able to survive for up to κ generations as described in algorithm 2. This notation further generalizes the old (μ, λ) and $(\mu + \lambda)$ notations, where the "," notation means $\kappa = 1$ and "+" notation $\kappa = \infty$.

To design an ES one must first select an appropriate representation for an individual and the most natural one is prefered in most cases, if all parameters are of one type (e.g. a real number) a simple vector will suffice, if the types are mixed, a tuple of vectors is required. This however causes an increased complexity of the variation operator.

As for design of the variation operator there are some guidelines that should be followed when designing it.

Algorithm 2 $(\mu/\rho, \kappa, \lambda)$-ES

```
1: initialize population $P^0$ with $\mu$ individuals
2: set age for each $p \in P^0$ to $1$
3: set $t = 0$
4: repeat
5:     $Q^t = \{\}$
6:     for $i \in \{1 \ldots \lambda\}$ do
7:         select $\rho$ parents $p_1, \ldots, p_\rho \in P^t$ uniformly at random
8:         $q = variation(p_1, \ldots, p_\rho)$ with age $0$
9:         $Q^t = q \cup Q^t$
10:    end for
11:    $P^{t+1}$ = select $\mu$ best (wrt. fitness) individuals from $Q^t \cup \{p \in P^t : age(p) < \kappa\}$
12:    increment age by 1 for each $p \in P^{t+1}$
13:    increment $t$
14: until stop criterion fullfilled
```

Reachability every solution should be reachable from any other solution in a finite number of applications of the variation operator with probability $p > 0$

Unbiasedness the operator should not favour any particular subset of solution unless provided with information about problem at hand

Control the operator should be parametrised in such way that the size of the distribution can be controlled (practice had shown that decreasing it as the optimal solution is being approached is necessary)

kovariance

[7] [5]

2.3.1 CMA-ES

TODO [8]

2.4 Evolutionary strategies as replacement for reinfocement learning

uvod

2.4.1 OpenAI Evolutionary Strategy

Compared to reinfocement learning using evolutionary strategies have the advantage of not needing a gradient of the policy performance. Also as the state transition function is not known the gradient can't be computed using backpropagation-like algorithm. Thus some noise needs to be added to make the problem smooth and the gradient to be estimable. Here is where reinfocement learning and evolutionary strategies differ, reinfocement learning adds noise in the action space (actions are chosen from a distribution) while evolutionary strategies add noise in the parameter space (parameters perturbed while actions are deterministic).

dimensionality, what and when better

- Evolution Strategies as a Scalable Alternative to Reinforcement Learning [9]
  - algorithm description
  - comparison with RL
  - paralellization
  - smoothing in action vs. param space
- Improving Exploration in Evolution Strategies for Deep Reinforcement Learning via a Population of Novelty-Seeking Agents [10]
  - novelty search
  - ratio of fitness and novelty and its effects

Chapter 3

Results and discussion

You should have a separate chapter for presenting your results (generated by the stuff described previously, in our case in **??**). Remember that your work needs to be validated rigorously, and no one will believe you if you just say that 'it worked well for you'.

Instead, try some of the following:

- State a hypothesis and prove it statistically
- Show plots with measurements that you did to prove your results (e.g. speedup). Use either `R` and `ggplot`, or Python with `matplotlib` to generate the plots.[1] Save them as PDF to avoid printing pixels (as in figure 3.1).
- Compare with other similar software/theses/authors/results, if possible
- Show example source code (e.g. for demonstrating how easily your results can be used)
- Include a 'toy problem' for demonstrating the basic functionality of your approach and detail all important properties and results on that
- Include clear pictures of 'inputs' and 'outputs' of all your algorithms, if applicable

It is sometimes convenient (even recommended by some journals, including Cell) to name the results sub-sections so that they state what exactly has been achieved. Examples follow.

[1]Honestly, the plots from `ggplot` look <u>much</u> better.

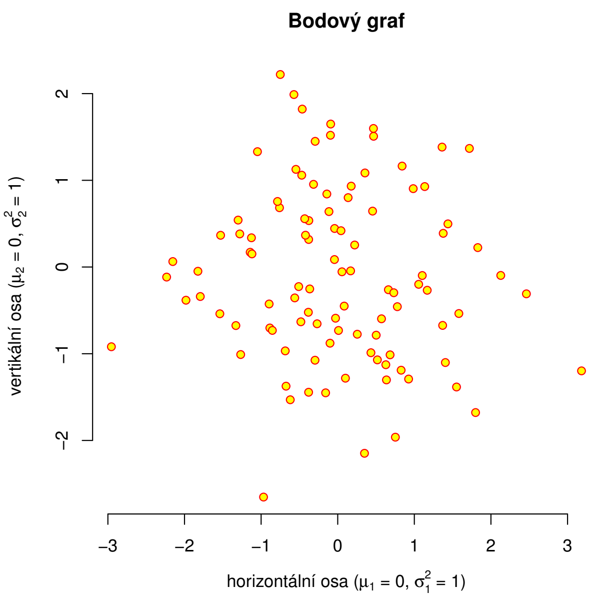

Figure 3.1 This caption is a friendly reminder to never insert figures "in text," without a floating environment, unless explicitly needed for maintaining the text flow (e.g., the figure is small and developing with the text, like some of the centered equations, as in ??). All figures *must* be referenced by number from the text (so that the reader can find them when he reads the text) and properly captioned (so that the reader can interpret the figure even if he looks at it before reading the text — reviewers love to do that).

3.1 SuperProgram is faster than OldAlgorithm

3.1.1 Scalability estimation

3.1.2 Precision of the results

3.2 Weird theorem is proven by induction

3.3 Amount of code reduced by CodeRedTool

3.3.1 Example

3.3.2 Performance on real codebases

3.4 NeuroticHelper improves neural network learning

3.5 What is a discussion?

After you present the results and show that your contribution works, it is important to *interpret* them, showing what they mean for the more general public.

Separate discussion sections are common in life sciences where ambiguity is common and intuition is sometimes the only thing that the authors have; exact sciences and mathematicians do not use them as often. Despite of that, it is nice to precisely set your output into the existing environment, answering:

- What is the potential application of the result?
- Does the result solve a problem that other people encountered?
- Did the results point to any new (surprising) facts?
- Why is the result important for your future work (or work of anyone other)?
- Can the results be used to replace (and improve) anything that is used currently?

If you do not know the answers, you may want to ask the supervisor. Also, do not worry if the discussion section is half-empty or completely pointless; you may remove it completely without much consequence. It is just a bachelor thesis, not a world-saving avenger thesis.

Conclusion

In the conclusion, you should summarize what was achieved by the thesis. In a few paragraphs, try to answer the following:

- Was the problem stated in the introduction solved? (Ideally include a list of successfully achieved goals.)
- What is the quality of the result? Is the problem solved for good and the mankind does not need to ever think about it again, or just partially improved upon? (Is the incompleteness caused by overwhelming problem complexity that would be out of thesis scope, or any theoretical reasons, such as computational hardness?)

This is quite common.

- Does the result have any practical applications that improve upon something realistic?
- Is there any good future development or research direction that could further improve the results of this thesis? (This is often summarized in a separate subsection called 'Future work'.)

Bibliography

[1] Hilary Glasman-Deal. *Science research writing for non-native speakers of English.* World Scientific, 2010.

[2] Donald Ervin Knuth. *TEX and METAFONT: New directions in typesetting.* American Mathematical Society, 1979.

[3] Leslie Lamport. *LATEX: a document preparation system: user's guide and reference manual.* Addison-Wesley, 1994.

[4] Don Sparling. *English or Czenglish? Jak se vyhnout čechismům v angličtině.* Státní pedagogické nakladatelství, 1989.

[5] Günter Rudolph. "Evolutionary Strategies". In: *Handbook of Natural Computing.* Ed. by Grzegorz Rozenberg, Thomas Bäck, and Joost N. Kok. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 673–698. ISBN: 978-3-540-92910-9. DOI: 10.1007/978-3-540-92910-9_22. URL: https://doi.org/10.1007/978-3-540-92910-9_22.

[6] P. A. Vikhar. "Evolutionary algorithms: A critical review and its future prospects". In: *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC).* 2016, pp. 261–265. DOI: 10.1109/ICGTSPICC.2016.7955308.

[7] Hans-Paul Schwefel and Günter Rudolph. "Contemporary evolution strategies". In: *Advances in Artificial Life.* Ed. by Federico Morán et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 891–907. ISBN: 978-3-540-49286-3.

[8] Nikolaus Hansen. *The CMA Evolution Strategy: A Comparing Review.* 2006.

[9] Tim Salimans et al. *Evolution Strategies as a Scalable Alternative to Reinforcement Learning.* 2017. arXiv: 1703.03864 [stat.ML].

[10] Edoardo Conti et al. *Improving Exploration in Evolution Strategies for Deep Reinforcement Learning via a Population of Novelty-Seeking Agents.* 2018. arXiv: 1712.06560 [cs.AI].

Appendix A

Using CoolThesisSoftware

Use this appendix to tell the readers (specifically the reviewer) how to use your software. A very reduced example follows; expand as necessary. Description of the program usage (e.g., how to process some example data) should be included as well.

To compile and run the software, you need dependencies XXX and YYY and a C compiler. On Debian-based Linux systems (such as Ubuntu), you may install these dependencies with APT:

```
apt-get install \
  libsuperdependency-dev \
  libanotherdependency-dev \
  build-essential
```

To unpack and compile the software, proceed as follows:

```
unzip coolsoft.zip
cd coolsoft
./configure
make
```

The program can be used as a C++ library, the simplest use is demonstrated in listing 1. A demonstration program that processes demonstration data is available in directory `demo/`, you can run the program on a demonstration dataset as follows:

```
cd demo/
./bin/cool_process_data data/demo1
```

After the program starts, control the data avenger with standard `WSAD` controls.

Listing 1 Example program.

```
#include <CoolSoft.h>
#include <iostream>

int main() {
  int i;
  if(i = cool::ProcessAllData()) // returns 0 on error
    std::cout << i << std::endl;
  else
    std::cerr << "error!" << std::endl;
  return 0;
}
```