Cover Letter

I have a specialization in Machine Learning and other related fields.
Also, I have developed a new **global optimization method that is 3 times better than Multistart** (without basins of attraction).
(https://medium.com/@pushkarevvaleriyandreevich/gradient-descent-that-we-must-have-5a4542e218a0
Read from What to do (with gradient descent)?)
That method performs **2-3 times better than Multistart**.

And if I add basins of attractions or other optimization techniques, I will outperform even GlobalSearch and others.

Also, I have developed **a communication technique that outperforms what existed in the Linux kernel** (with a few security issues at the time).
50 million calls per second versus 5 million calls. Or more than 10 times.
(**Part 1 - Library and main concepts**:
https://medium.com/@pushkarevvaleriyandreevich/making-libs-drivers-verilog-endpoints-for-custom-hardware-for-windows-linux-f8cf2d1e8efe)
(**Part 2 - Hardware endpoint**
https://medium.com/@pushkarevvaleriyandreevich/making-libs-drivers-verilog-enpoints-for-custom-hardware-for-windows-linux-8a6f580aa3f3)

Yes, I hear that **about 3–10% improvement in one discipline is enough for a PhD**, not 2-10 times).

Also, I have a novel NLU approaches that looks promising.

Why I'm out of programming.

The basis is simple - nowadays we (humans) even have software that can program better than us (DreamCoder (DARPA)).

Also, all translation from Lisp to any language (C#/C++/Rust or even C++) can be done with ChatGPT or analogs.

So I can say that within 2-5 years there will be no any programmer vacancy (at least in non-specialized cases).

What I plan to build?

I plan to finish my IPC generator and add security and other.

Also I want to implement my Global Optimization on GPU (there is no scientifically task without heavy computations on GPU).

In short - A global optimization pipeline with filters based on simulators and some text heuristics.

I have better global optimization algorithm so I can spend less time to grow a tissue variants.

And I want to build an Ideal Optimization Pipeline.

First, simply running global optimization is cost-sensitive, we have too many solutions, and checking all of them is almost impossible.

But we can:

1) **Add some approximations of a solution** (with a neural net that is trained to act like a bloom filter or even an SVM (that can act like a set of rules)).

2) **Make many bloom filters** to reduce the possible solution space.

3) Run developed global optimization within that area (that is already 2-3 times faster)

And that will work better than GlobalSearch from Matlab (and other algorithms):

1) because we have not only basins of attractions but also approximations based on neural nets

2) We have a better global optimization algorithm beneath all of that

3) We can even add some NLU features to make more conditions/filters

Also, I must transfer all of that to OpenCL/Tensorflow to make it usable.

And I can solve any problem, even within computational engineering, faster than others.

What about NLU?

Well, I have a modern NLU approach that covers almost all needs and can be verified and explained.

Heuristics, constrains and rules can greatly decrease the number of possible solutions during any type of work.

Also, nobody will build a model(s) of objects in Diagram Editor or so on (try to build 2-3 things with more than 20 parts) on a regular basis.

First,

What about objects and the properties of objects?

Again, there are many benchmarks for Q&A. And nowadays, neural networks with a 30 mb size can find answers better than humans.

And with Q&A it's possible to get almost any information about objects (what is the **color** of **object_name**).

It takes two modules to do that - the first module projects objects to the syntax tree:

{book {num_of_pages = 10, color = white, author=null}} - white book with 10 pages.

The second one - builds a question to properties

{book {num_of_pages = 10, color = white, author=null}} - Who is the author of a white book with 10 pages?

Note that we can ask many questions, and even incorrect ones (that will lead us to incorrect results), so we need to add some attributes (to define the right question for that item). (Whenwolf)

After all of that is completed - we can extract data about all defined objects and their properties.

So defining the simplest generator of questions based on json-like object descriptions is a trivial thing. (That's called reading :))

And we don't need to train a transformer with 20 billion parameters to get information about some objects (that is the part of filters).

We can extract any data about objects and states with questions.

Is that new/how it differs from existing NLU methods?

Almost all information about any objects can be extracted. No need in another 20b parameters transformer/etc.

Any special cases (that away from Natural Language) can be stored in a separate storage (HashTable or transfrormer).

What with rules and so on?

Rules (constraints), heuristics, and even functions - all objects too, but not in physical world (that can have one or more objects as arguments, and change existing objects or add new ones :)).

And we can get all needed information about that objects from text with Q&A.

So, again, I must add several abstractions/functions that represent physical space and other needed properties of the constructed object.

And I can get all the information through Q&A.

If I don't want to implement many functions by hand, I can use simulators based on Machine Learning models (text, functions, and anything else),

or specialized simulators of physics(differential equations)/chemistry and so on.

For example - it's about 10 basic functions for common physical world (coordinates, rotations, transformations, sets operations).

More complex functions can be constructed from basic functions (even with the help of DreamCoder and few-shot learning).

What with levels of abstractions?

Abstraction can be represented as Sets of objects that translates with some function to another object (and back). Directly with properties mapping or in a complex way.

If we can define translation functions with a few examples (or use approximation based on machine learning) it's not hard to build a hierarchy of abstraction levels.

And define heuristics/constrains and result properties on any level of abstraction.

What will I get?

System that can find a solution with text-based rules faster than any other tool.

With the abstractions, large library of objects and fast generation of rules/heuristics I can get a very useful tool that can cover almost all automation design needs.

A perfect tool for prototyping (no one will rewrite 100 rules to generate something).

And even work (you can always check what the rules are for our tool, not a "Magic of Neural Nets" or 1001 file with **favorite language**/grammar).

Why can I do that?

Well, I have 5 years of higher education and about 4 years as a programmer in top Russian firms.

Also, I have already developed a new Global Optimization algorithm.

Resume

Pushkarev Valeriy Andreevich

Male, 32 years, born on 14 February 1990

Contacts   +7 (951) 392-73-50

PushkarevValeriyAndreevich@proton.me — preferred means of communication

Novosibirsk, willing to relocate, not prepared for business trips

Middle C# Developer

Specializations:Programmer, developer

Employment: full time

Work schedule: full day

Work experience 5 years

January 2021 — currently

1 year 8 months

FreelanceC# Backend developerWWF, FPW, .net core developer

October 2019 — april 2020

7 months

Siberian Networks

www.sibset.ru/Telecommunications, Communications

Middle c# developerDevelopment and testing of an ETL system for integrating a new BPM, development of a text template for generating classes according to Oracle tables (blToolkit). DeadLock-Free architecture due to diversification into only 2 consecutive tasks (Loading-preparing objects and validating-unloading)

Loaded data from all systems.CRM has not yet begun to be introduced.

Support for the old CRM (viewing and optimizing query plans, adding indexes, archiving obsolete data)

January 2018 — april 2018

4 months

Kaspersky Labwww.kaspersky.ru

IT, System Integration

MIddle C# Test Automatisation EngineerTest refactoring.Connected Selenium.Brought Selenium and UI Automation entities to single classes.Made a single service, brought everything to a single interface.Increased the speed of tests by 2.5 times.Test support.

April 2016 — october 2016

7 months

GIS2gis.ru

IT, System Integration, Internet

Middle c# developerRefinement of the UI of the BuildMan system, adding notifications via RabbitMQ and WCF (microservice), Adding a search for the state of builds by various criteria. Debugging the server side - transferring WWF from a custom provider to get workflow data to SQL, eliminating many Race-Conditions due to incorrect use of WWF.

February 2015 — march 2016

1 year 2 months

BCS Limited

Junior C# developer

Support for an ETL system for integrating and aggregating transaction data. Development of a new MDM solution (according to SOLID) and transfer of aggregation from SQL sheduler scripts to C#. Database size 1.5 TB, more than a million transactions per hour.Reporting development,

Key skills

Quick learning

Driving experience

Driver's license category B

About me

Can solve problems like this:https://github.com/m4rs-mt/ILGPU/issues/639)If you have doubts about The Hardware/Software Interface от Washington University (from asm to C) - i can send pdf)

Have many certificates in AI, Electronics, and so from Washington University, Berkley и MIT with distinction.Even have Microsoft 98-361 certificate

Higher education

2013North-Eastern Federal University, NEFU, Yakutsk Electronics, Automatics and Electric drives design engineer

Electronic certificates

Stanford Machine learning

Berkley CS169.1x: Software as a Service

MITx 6.002x: Circuits and Electronics

Stanford cryptography I

Washington university The Hardware/Software Interface

Berkeley CS188.1x: Artificial Intelligence

Transcripts of all degrees

Higher education -

Institute:

Federal State Autonomous Educational Institution of Higher Education "M. K. Ammosov North-Eastern Federal University"

Specialization:

Electric drive and automation of industrial installations and technological complexes

(5 years total)

Disciplines:

Discipline Name:	Total Hours:	Final Grade(ECTS):
1. English language	342	A
2. Physical culture	408	A
3. Domestic history	125	B
4. Philosophy	125	B
5. Russian language and culture of speech	115	Pass
6. Jurisprudence	60	Pass
7. Economic	87	Pass
8. Mathematics	700	A
9. Computer science	300	Pass
10. Physics	500	A
11. Chemistry	150	B
12. Ecology	70	Pass
13. Physical foundations of electronics	172	Pass
14. Theoretical mechanics	180	Pass
15. Descriptive geometry. Engineering graphics	192	Pass
16. Materials science. Technology of structural materials	140	B
17. Mechanics	180	A
18. Theoretical foundations	340	A

of electrical engineering		
19. Electrical machines	170	B
20. Electrical and electronic devices	170	B
21. Metrology, standardization and certification	70	D
22. Life safety	180	Pass
23. Electric drive	150	B
24. Culturology	135	Pass
25. Sociology	135	Pass
26. Mathematical modeling in electrical engineering	80	Pass
27. Fundamentals of programming	70	Pass
28. Electronic and microprocessor technology	60	B
29. Circuit design of electric drive control systems	75	A
30. Introduction to the specialty	30	Pass
31. Introduction to electrical engineering	45	A
32. Psychology of business communication	70	Pass
33. Foreign language in the field of professional communications	200	Pass
34. Automation of a physical experiment	200	Pass
35. Theory of electric drive	220	A
36. Control systems of electric drives	300	A
37. Elements of automation systems	150	B
38. Automated electric drive of standard	150	B

production mechanisms		
39. Economics and organization of production of electric drives	100	B
40. Theory of automatic control	150	B
41. Converter technology (power electronics)	130	B
42. Power supply of industrial enterprises	150	A
43. Microprocessor tools in electric drives and technological complexes	130	A
44. Electrical equipment and automation of production facilities	130	A
45. Modeling of electric drives and automation systems	170	A

Total hours: 7806

Online certificates

Course Name	Duration	Description	Grade	Link to certificate
6.002x: Circuits and Electronics	4 month	All basic electronics - from resistors to kirgoph polynomial, complex circuits and electromagnetic receivers/transfers	Pass	https://s3.amazonaws.com/verify.edx.org/downloads/ea9314d9749142d5951369866a692cb8/Certificate.pdf
Stanford Machine learning	4 month	Machine Learning By Andrew Ng This is reissued certificate. Now on Coursera there are about 3 courses each 3-4 week long. All basic thing about machine learning.	96.24%	https://coursera.org/share/4bc6e73fa9ecaa7fea29f01f0089c494
Stanford cryptography I	3 month	All basic cryptography from Caesar down to block ciphers, PGP, and all basic attacks.	100% with Distinction	https://coursera.org/share/fbeae640fad1de29171275777c6531db
The Hardware/Software Interface	3 month	This undergraduate course covers basic principles of the hardware/software interface including hardware architecture, memory hierarchy, x86 assembly programming, and C vs. Java concepts	83.09% with Distinction	No link to certificate (
Berkley CS188.1x: Artificial Intelligence	2 month	MDP, BFS, DFS, A* and all other basic concepts of space exploration and agent building.	Pass	https://s3.amazonaws.com/verify.edx.org/downloads/f2ee5fa119164fbab3b1e97747b9d0d9/Certificate.pdf

Representative publications (on medium):

https://medium.com/@PushkarevValeriyAndreevich

1. Global optimization better than multistart in MatLab (without basins and so on)
2. Making Libs, Drivers and verilog endpoint generator based on vDSO (SMO) - call without syscall, or 50M messages per second on 4 cores. (better that anything in Linux core)
3. Making Libs, Drivers and verilog endpoint - makind data dependancy graph and multi-issue on hardware enpoint, make levels of executions and methods endpoints

https://medium.com/@PushkarevValeriyAndreevich2

1. Making selector that takes for 20% less space than in ucdavis