

On Generation of New Ideas for PhD Research in Computer Science and Engineering: Concepts and Explanations

Table of Contents:

1. Introduction
2. Other Relevant Studies (10)
3. Classification, Explanation, and Some Popular Examples (10)
4. Examples from Turing Awards (10)
5. Examples from the List of Top 500 Computer Scientists of the World,
maintained at the UCLA
6. Conclusions
7. Classified References

Version: 28.2.2012. @ 10am.

1. Introduction

This paper represents an effort to help PhD students in computer science and engineering to generate good original ideas for their PhD research. Our effort is motivated by the fact that most of PhD students demonstrate an eager desire to get guided by an appropriate methodology, as indicated in [X]. Limited duration of PhD studies at most universities all around the world impose a requirement of efficient PhD research process, which encourages positive experience in generating ideas for research to be considered and applied.

2. Related Studies

This paper is a follow up of four previous papers on different aspects of research generation methodology [Milutinovic95 and Milutinovic2008] and research presentation methodology [Milutinovic96 and Milutinovic2012].

In addition, this paper builds on the top of 10 other representative studies related to methodologies for research innovation in science and engineering [Cummings2005, Faulkner94, Livari2007, Kline86, Libarkin2002, Linn87, Pavon2005, Perkmann2007, Swan97, Wang2003, Dorfler2011, DorflerBaracskaei2010, Dorfler2012, DorflerAckermann2012], and tries to transform existing innovations into an original set of 10 different methodological approaches to innovation in computer science and engineering.

The next section sheds more light on 10 different idea generation methods introduced in this paper. Examples are given from a course on research innovation taught for several years now at the University of Belgrade (papers referenced in the next section are the paper analyzed in the mentioned course).

The follow up section gives some numerics generated using the Google Scholar System. These numerics have to be taken conditionally, since they are based on the individual subjective analysis of the most referenced papers, by a given individual (award laureate), or in a given field (teaching topic). However, the main goal of this paper is to help PhD students to generate good new ideas, so the numerical examples are only for the illustration purposes.

3. Classification

This section, together with Figure ABCDE, defines the essence of 10 different methods that could be used to generate new R+D ideas in computer science and engineering. The methods are not orthogonal, so any innovation can belong to one or more of the methods. These methods are meant to serve only as guidelines for PhD students.

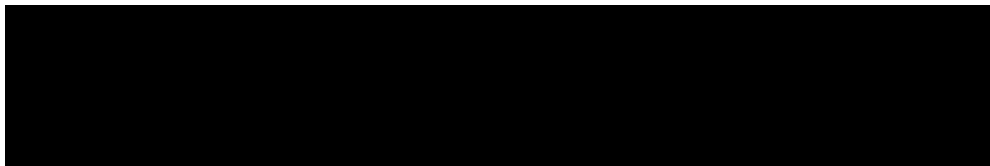
3.1. Mendeleyevization: Inductor versus Catalyst (M1 vs M2)

Definition:

If one of the classification classes includes no examples, it first has to be checked why is that so. If it is so because it makes no sense, an appropriate explanation is in place. If it is so because the technology or the applications are not yet ready for such an approach, one can act in the same way as the famous chemists Mendeleyev: Empty positions in any classification are potential avenues leading to new inventions. We refer to such an approach as: Mendeleyevization.

Graphical Presentation:

As indicated in Figure A, these inventions sometimes need an inductor (a resource that makes an invention happen) or a catalyst (a resource that turns a known invention without potentials into an invention with potentials). Both, inductor and catalyst, can be a hardware or a software resource.



A: Mendeleyevization (Inductor versus Catalyst) – M1 vs M2

Examples:

As far as M1/M2, the famous classification of computer systems by Mike Flynn (SISD, SIMD, MISD, MIMD) initially included no examples of the MISD type. Later on, a DFT machine (generated using the M1 method) was categorized as an MISD machine [Milutinovic86A], as well as one pipelined machine (generated using the M2 method), namely [Milutinovic87C]; the DFT served as an inductor, and pipeline as a catalyst. Other popular examples are related to various signal processors and process accelerators.

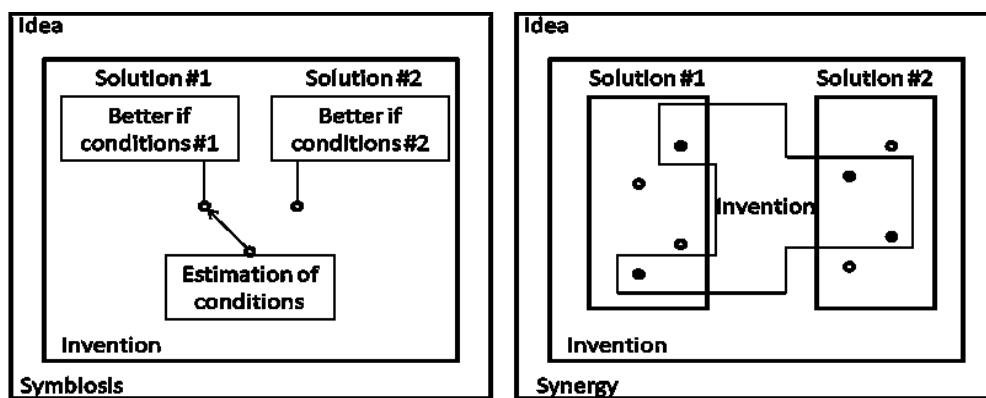
3.2. Hybridization: Symbiosis versus Synergy (H1 vs H2)

Definition:

Sometimes two classification classes can be combined, in order to obtain a hybrid solution (hybridization). Hybrid solutions can be symbiotic (measuring the conditions in the environment and switching from one approach to the other, so that each approach is active all the time while the conditions are such that it provides better performance compared to the other approach) or synergistic (creating a new approach, which, for each particular solution element takes the better solution element of two different approaches).

Graphical Presentation:

Essence of Hybridization is shown in Figure B. The assumption here is that one solution is better under one set of conditions, and the other solution is better under another set of conditions. Another assumption is that solution elements of one solution are better in one domain and that solution elements of another solution are better in another domain. Consequently, the complexity of a new solution generated using the H1 method is always higher than the complexity of each existing solution used to generate the hybrid solution, while it may not be the case when the H2 method is used; however, it is easier to develop ideas based on H1, compared to H2, but often times it is easier to implement solutions based on H2, rather than H1.



B: Hybridization (Symbiosis versus Synergy) – H1 vs H2

Examples:

As far as H1/H2, the essence of [Milutinovic85] is that two algorithms are combined into one on the either-one-or-the-other basis (using the H1 method), and on a combine-inherent-details basis (using the H2 method) in [Milutinovic87B]. Other popular examples include hybrid computers or computers that use special purpose accelerators, when appropriate data/process patterns are located.

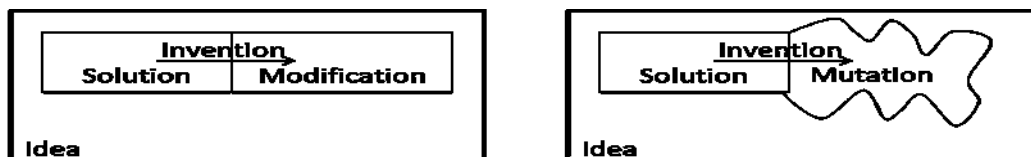
3.3. Transdisciplinaryrization: Modifications versus Mutations (T1 vs T2)

Definition:

Often times, good new ideas get generated if algorithms, procedures, ways of thinking, are ported from one field to another field, along the lines of transdisciplinary research methodologies (transdisciplinaryrization).

Graphical Presentation:

As indicated in Figure C, for an idea to work better in the new field, either smaller modifications or larger mutations have to be introduced. Modification means that the solution is ported directly, and that only the interpretation of related variables is different. Mutation means that just an analogy is used to generate a new idea from some initial solution, but realistically the new idea has nothing to do with the initial solution.



C: Transdisciplinaryrization (Modification versus Mutation) – T1 vs T2

Examples:

As far as T1/T2, [Milutinovic86B] ports algorithms from Silicon to GaAs (using the T1 method), and introduces appropriate modifications along the process, while [Milutinovic87A] creates a proposal for a novel computer architecture (using the T2 method), along the analogies with a biological honeycomb. Other popular examples include porting of the FFT from seismic signal processing to speech signal processing, or introduction of mathematical neural networks inspired by biological neural networks.

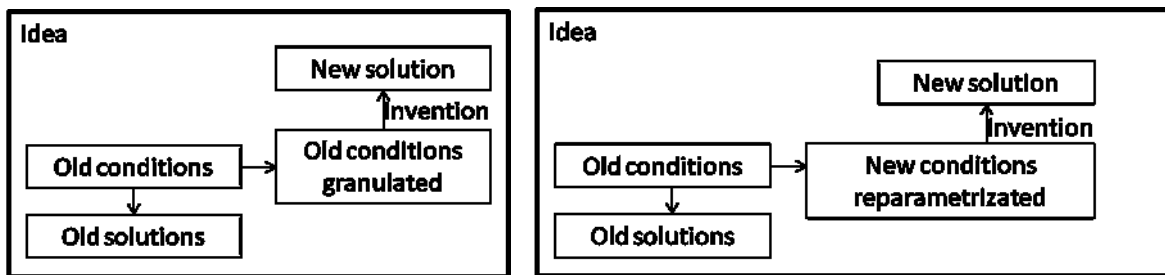
3.4. Remodelling: Granularization versus Reparametrization (R1 vs R2)

Definition:

Sometimes it is simply the best to take a research direction different (even opposite) compared to what others take (retrajectorization using remodeling). The different (opposite) research direction makes sense either if a more detailed set of parameters is in place (granularization, due to model changes because of application changes), or because parameters of the environment have changed permanently (reparametrization, due to model changes because of technology changes), as indicated in Figure D. The two alternatives are referred to as granularization and reparametrization.

Graphical Presentation:

Figure D indicates that better solutions can be generated, either if one switches to a more granular model (and sees something that could not be seen in the previous model that was coarser), or if one switches to a different model, if that is justified by changes in the environment, application, or technology (both evolutionary and revolutionary changes do happen, and it is wrong, after the change happens, to continue with solutions that were considered optimal prior to the change).



D: Remodeling (Granularization versus Reparametrization) – R1 vs R2

Examples:

As far as R1/R2, [Milutinovic88] offers a new algorithm (using the R1 method) that makes sense if an environment is represented with a more detailed model, while [Milutinovic89] offers a new solution in a changed environment (using the R2 method), when a design has to be ported from Silicon to GaAs (from the performance/price point of view, the best adder design for Silicon, the carry-lookahead adder, is among the worst ones for GaAs, and the opposite: the worst adder for Silicon, the ripple-carry adder, is among the best ones for GaAs; this is because GaAs gate delays depend on fan-out and ripple-carry adders are characterized with only the minimal fan-out, while the fan-out of the carry-lookahead adders depends on the word size, and can grow infinitely). Other popular examples are related to concept modeling in AI based on graphical representation with icons (in a model with fewer icons, one can make a conclusion which is different, and often times even opposite, compared to a conclusion made from a model with a much larger number of icons); also, when the environment changes (for example, the ratio of processing speed to communication speed changes), a different type of supercomputing network becomes optimal.

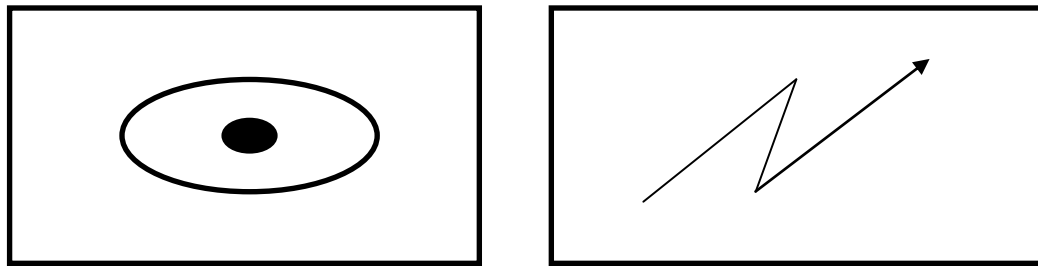
3.5. Unorthodoxization: *ViewFromAbove* versus *ViewFromInside* (U1 vs U2)

Definition:

This category encompasses the approaches that are difficult to classify (Figure E): Sometimes one sees something that others did not see for decades or centuries (*ViewFromAbove*) or one gets stroked by an idea of a genius with no ground in existing research (*ViewFromInside*).

Graphical Presentation:

Symbols in Figure E were chosen to create associations about the bird's view and the worm's view, and to induce creative thinking of the top-down or the inside-out approaches.



E: Unorthodoxization (*ViewFromAbove* versus *ViewFromInside*) – U1 vs U2

Examples:

As far as U1/U2, [Milutinovic2000] generated an innovation after trying to make a holistic view (U1), and [Milutinovic2001] introduces an idea after an effort is made to understand the intrinsic details of the problem (U2). Other popular examples include the contributions of Nobel Laureates Martin Perl and Jerome Friedman.

4. Examples from the Turing Award

The Turing award is treated as “the Nobel Prize of Computing”. For the first 10 single recipients, in the text to follow, we paraphrase the contribution for which they were awarded, and based on our study, we label each contribution as far as the innovation method that prevailed (see the mnemonic in the parenthesis). A summary is given in Figure 5.1.

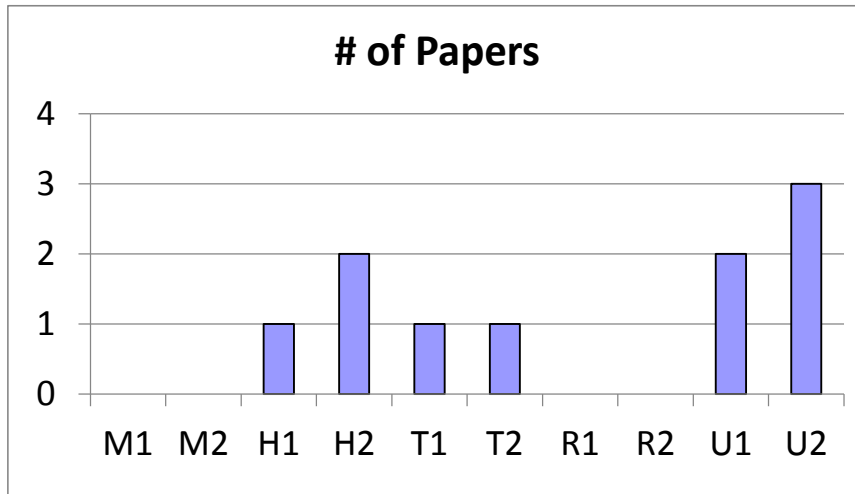


Figure 5.1. Number of Turing Awards based on the given innovation method.

In 1966, Alan J. Perlis was awarded for his influence in the area of advanced programming techniques and compiler construction. He was taking parts of the existing body of knowledge, and combining them into meaningful new entities, which could be classified as H1 method.

In 1967, Maurice V. Wilkes was awarded as the builder and designer of the EDSAC, the first computer with an internally stored program. As the new idea was born after a closer look was taken into the essence of the basic computing steps, this kind of research belongs to U2 category.

In 1968, Richard Hamming was awarded for his work on numerical methods, automatic coding systems, and error-detecting and error-correcting codes. A sequence of bits was looked at from a distance and protection patterns become clearly visible, so the research can be also classified as U1.

In 1969, Marvin Minsky was awarded for artificial intelligence. Details of the code leading to inference were studied, and patterns of interest for a new programming paradigm suddenly became visible, which brings one more invention in the U2 category.

In 1970, James H. Wilkinson was awarded for his research in numerical analysis to facilitate the use of the high-speed digital computer. Hence, the existing body of knowledge was moved into a new environment, together with appropriate changes, to make the computing faster, which represent an example of T1 accomplishments.

In 1971, John McCarthy was awarded for the lecture “The Present State of Research on Artificial Intelligence”. This research belongs to the same category as the research of Minsky (U2), except that the focus was somewhat different.

In 1972, Edsger W. Dijkstra was awarded as a principal contributor in the late 1950s to the development of the ALGOL, a high level programming language which has become a model of clarity and mathematical rigor. The language constructs were taken from the machine level to the high-language level, and along that path were combined and modified for a better match with the human way of thinking, which can be understood as a H2 research contribution.

In 1973, Charles W. Bachman was awarded for his outstanding contributions to database technology. Architectural constructs from the main memory level were ported to an upper level, and were mutated for a better match with technology requirements intrinsic to the upper level, so his research method belongs to T2 category.

In 1974, Donald E. Knuth was awarded for his major contributions to the analysis of algorithms and the design of programming languages, and in particular for his contributions to “The Art of Computer Programming” through his well-known books. Following a wholistic view, it was determined what are the alternatives for the basic algorithms of computing, so once again the U1 methodology was used.

In 1977, John Backus was awarded for profound, influential, and lasting contributions to the design of practical high-level programming systems, notably through his work on FORTRAN. He used a similar research pattern as Dijkstra, except that the application of the created HLL was somewhat different, so his work can be interpreted as of H2 type. .

5. Examples from the Nobel Laureate Research

This section classifies 10 different Nobel Prize achievements and classifies them into the 10 idea generation methods introduced in this paper. The summary is given in Figure 4.1.

Innovation in [Arrow] seems to have been created predominantly along the lines of the method U1. Likewise, innovation in [Cooper] corresponds to H, innovation in [DeGennes] to M2, innovation in [Friedman] to H1, innovation in [Glashow] to U1, innovation in [Kroto] to U2, innovation in [Maskin] to T1, innovation in [Perl] to U2, innovation in [Richardson] to T2, and innovation in [Wilson] to U1.

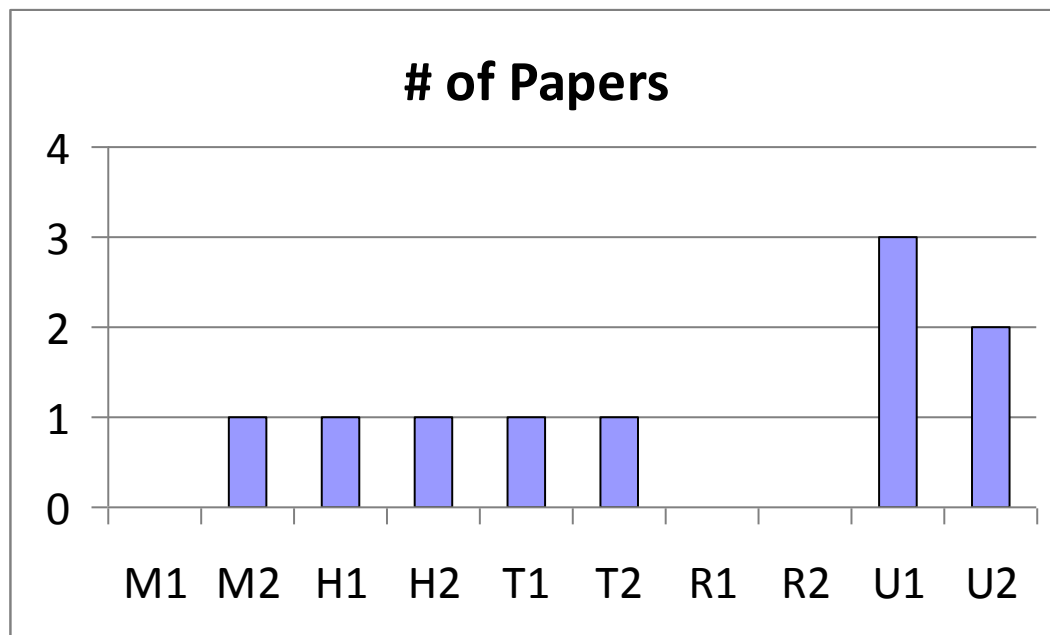


Figure 4.1. Number of Nobel Prizes based on the given innovation method.

6. Numerical Values from Google Scholar (distribution of 10 methods in top 10 papers of 10 different fields)

This section presents the distribution of presented innovation methods in the top 10 most referenced research papers (excluding surveys and tutorials, as well as papers on simulation-based or mathematics-based comparisons of existing approaches) for 10 different research fields, as determined by the authors of this study.

An author opinion analysis for the field of information systems (Google: “information systems”) gives the distribution is presented in Figure 7.1. For the field of optimizing compilers (Google: “optimizing compilers”) the distribution is presented in Figure 7.2. For the field of database technology (Google: “database engineering”) the distribution is presented in Figure 7.3. For the field of interconnection networks (Google: “interconnection networks”) the distribution is presented in Figure 7.4. For the field of object oriented programming (Google: “oriented programming”) the distribution is presented in Figure 7.5. For the field of computer architecture (Google: “computer architecture”) the distribution is presented in Figure 7.6. For the field of artificial intelligence (Google: “artificial intelligence”) the distribution is presented in Figure 7.7. For the field of performance evaluation (Google: “performance evaluation”) the distribution is presented in Figure 7.8. For the field of computer graphics (Google: “computer graphics”) the distribution is presented in Figure 7.9. For the field of multiprocessor systems (Google: “multiprocessor systems”) the distribution is presented in Figure 7.10.

The overall conclusion is, for the theory oriented research, methods U1 and U2 prevail; for the practice oriented research, methods H1, H2, T1, and T2 prevail.

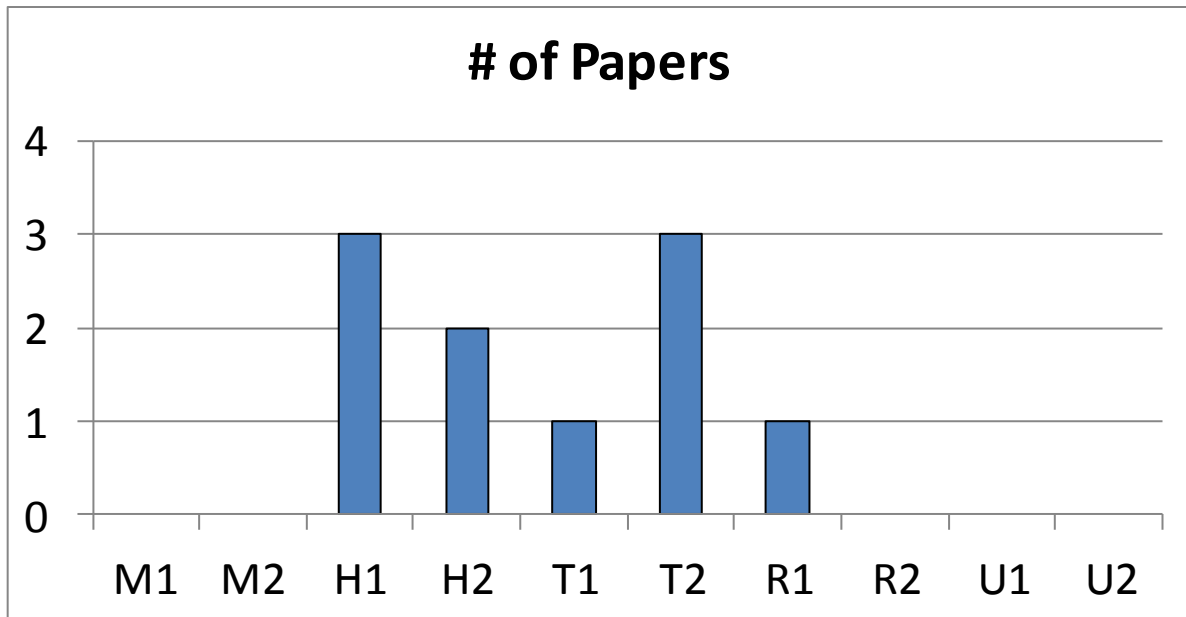


Figure 7.1. Distribution of innovation methods for “information systems”

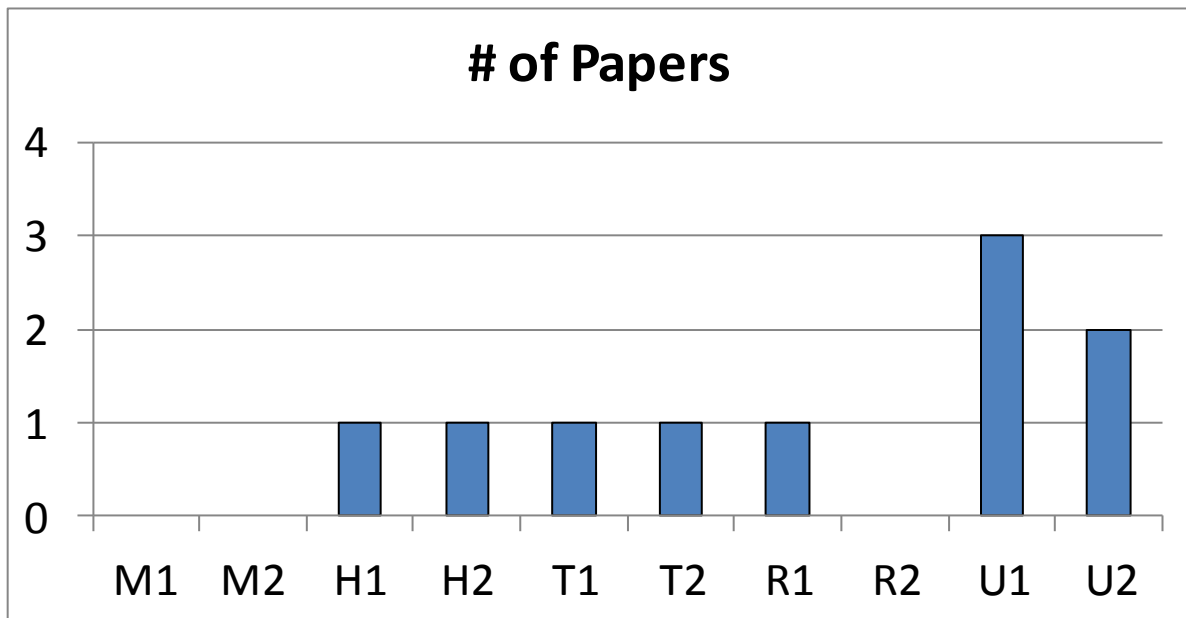


Figure 7.2. Distribution of innovation methods for “Optimizing Compilers”

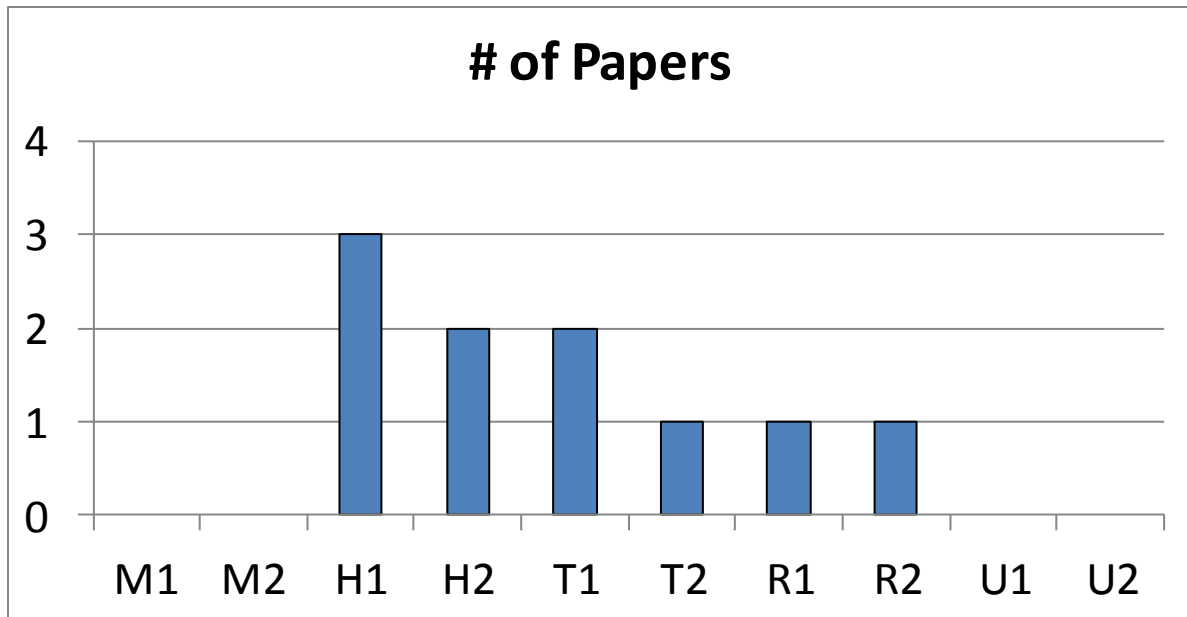


Figure 7.3. Distribution of innovation methods for “Database Technology”

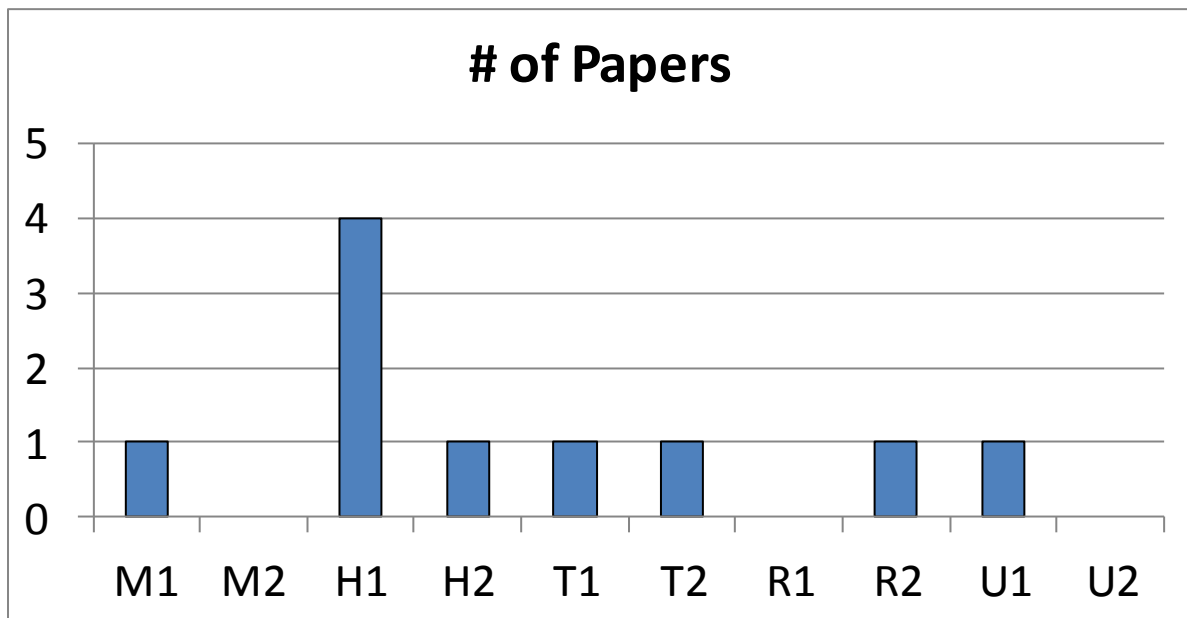


Figure 7.4. Distribution of innovation methods for “Interconnection Networks”

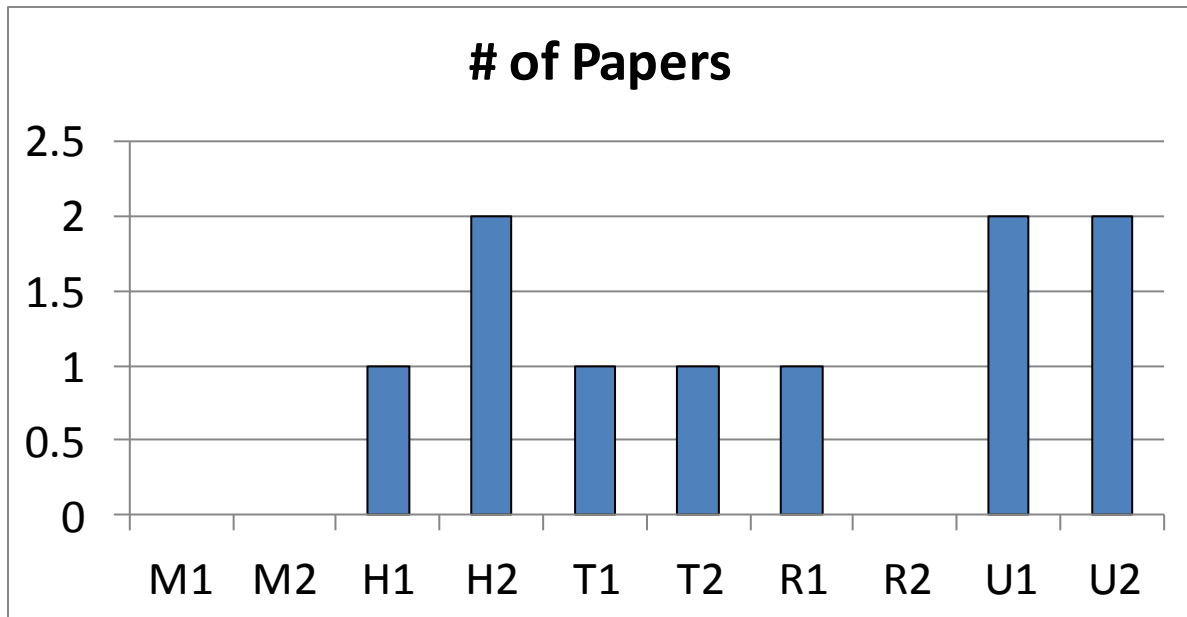


Figure 7.5. Distribution of innovation methods for “Object Oriented Programming”

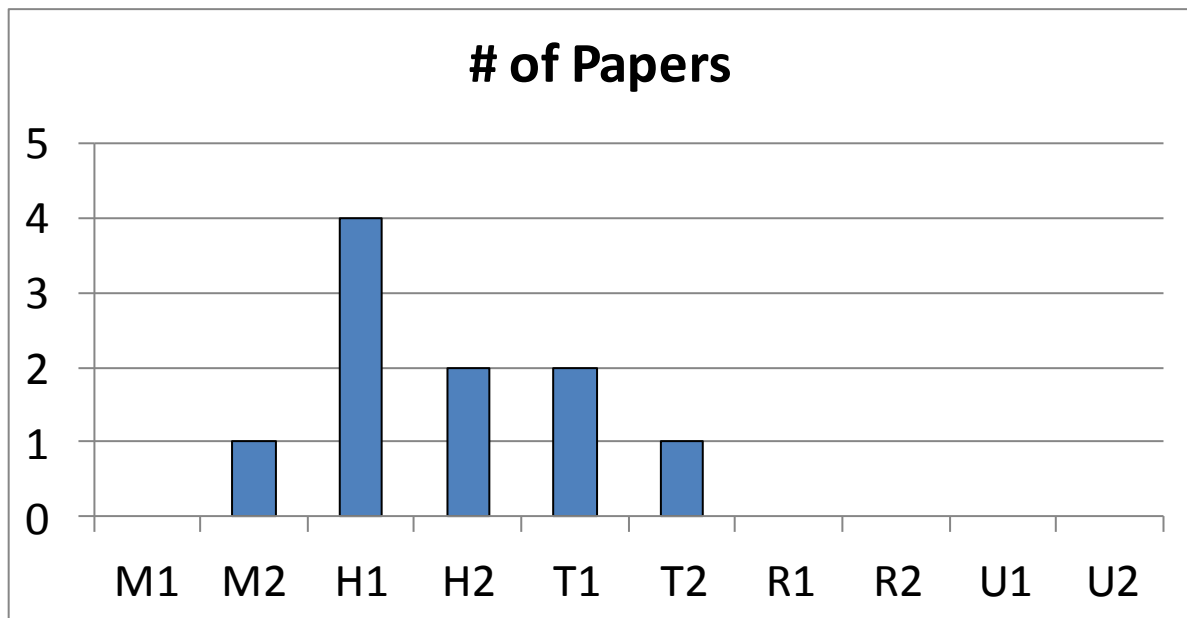


Figure 7.6. Distribution of innovation methods for “Computer Architecture”

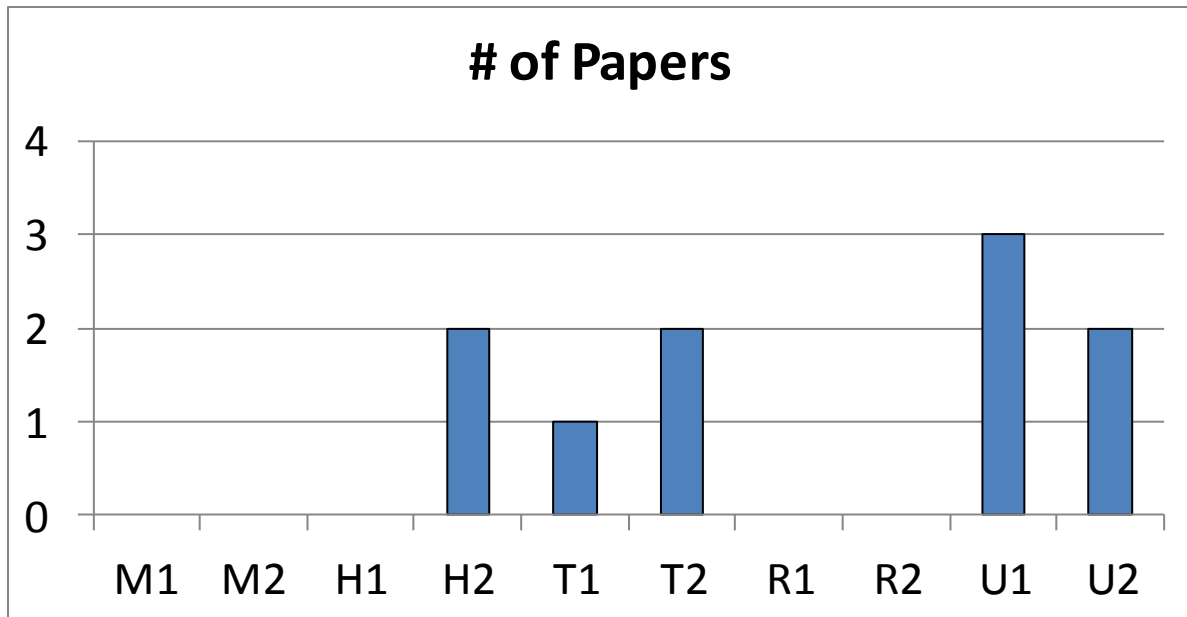


Figure 7.7. Distribution of innovation methods for “Artificial Intelligence”

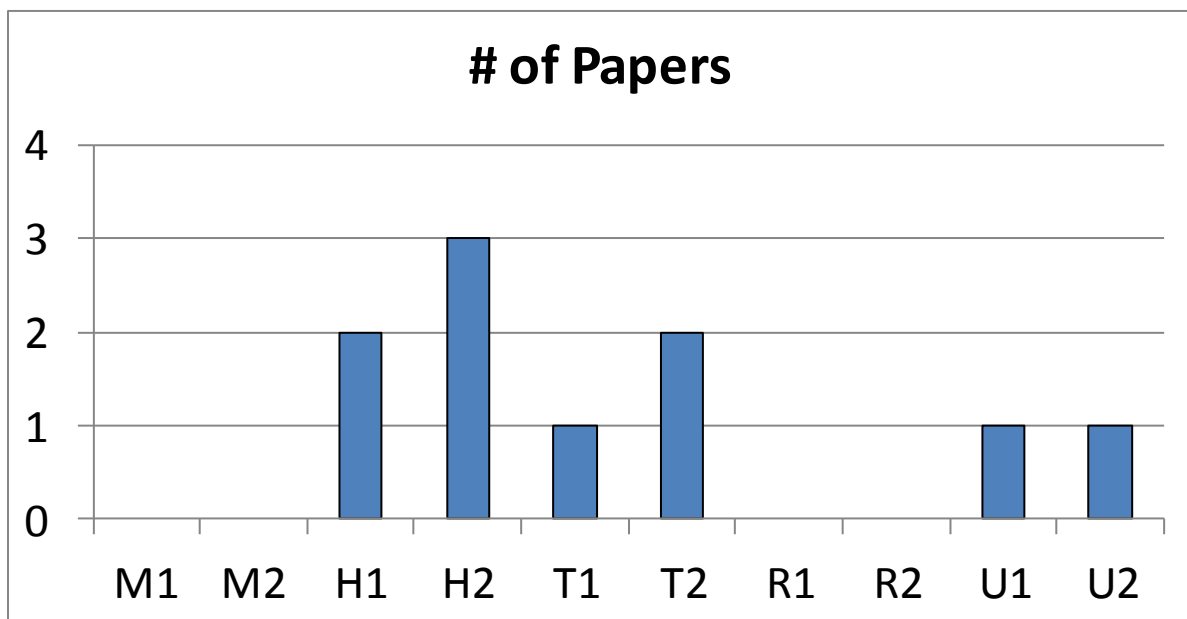


Figure 7.8. Distribution of innovation methods for “Performance Evaluation”

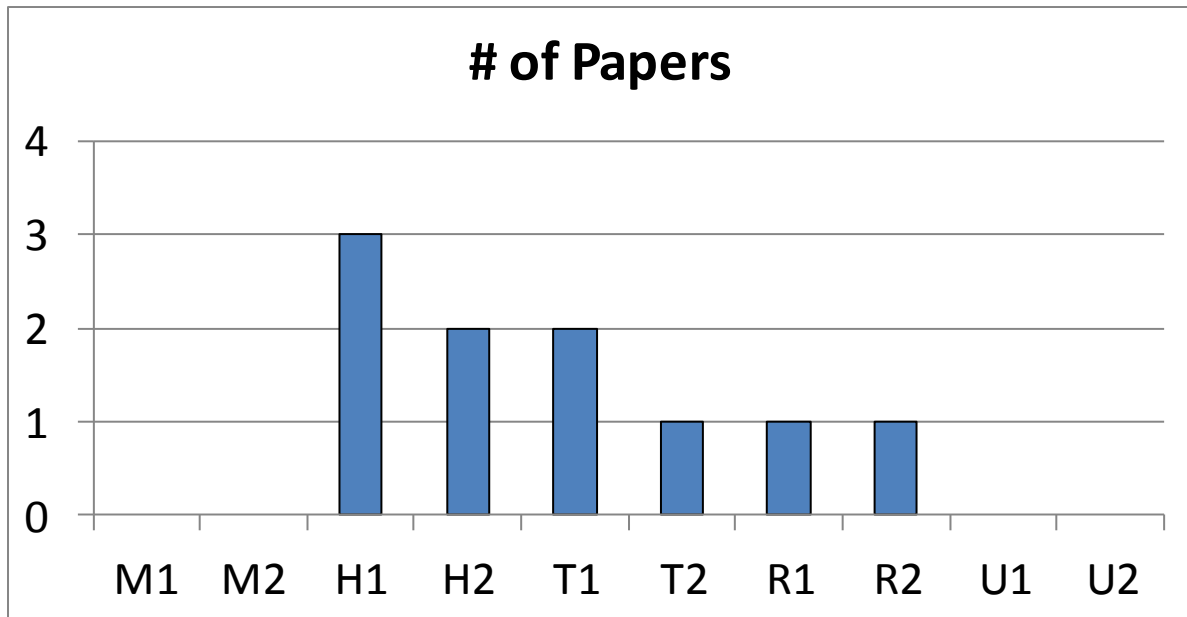


Figure 7.9. Distribution of innovation methods for “Computer Graphics”

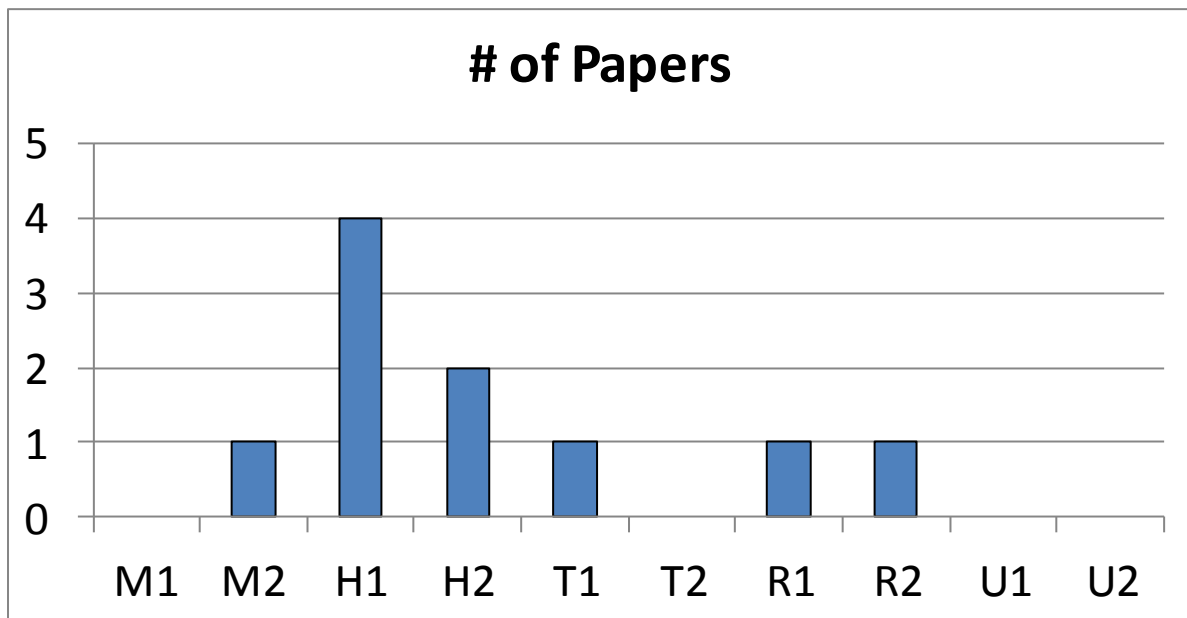


Figure 7.10. Distribution of innovation methods for “Multiprocessor Systems”

7. Conclusions

This paper introduces and explains 10 different methods that one can use to generate ideas for PHD research. It also analyses: (a) the most successful research efforts (of the first 10 individual laureates of the Turing Awards), and (b) the examples of the Ph.D. research of 10 authors of this paper.

This study is of benefit to PhD students who are eager to generate original ideas that could serve the basis of their PhD research. The presented methodology implies that the PhD student is first asked to create a survey of existing solutions to the problem attacked by his/her PhD research. The classification may include classes without examples, which opens doors for methodologies M1/M2. Algorithms/approaches inherent to various examples in different classes can be combined, which opens doors for methods H1/H2. Further on, with less or more modifications, one can port algorithms/approaches from one field to other fields, which opens doors for methods T1/T2. If conditions related to the understanding of the environment change, or our perception of the environment changes, doors get open for methods R1/R2. Finally, a bird's view or the worm's view of the surveyed examples may bring improvements along the lines of methods U1/U2.

Future work on this subject implies that the conclusions generated by the authors of this paper (as far as what research innovation belongs to what idea generation method) be verified from the authors of the original papers from the list of the most referenced ones. Also, as the experiences are gained in the work with PhD students, a follow up paper (maybe a decade later) could summarize new findings related to the advisory work with young talents using the idea generation methodologies advocated in this paper.

8. Classified References Used in the Educational Process

A course at the University of Belgrade teaches the methodology of this research to undergraduate and graduate students, and gives them 10 handouts (10 papers based on the 10 innovation approaches presented here). These 10 papers are listed below.

M1: Mendeleyevization/Inductor

[Milutinovic86a] Milutinovic, V., Fortes, J., Jamieson, L.,
A Multiprocessor Architecture for Real-Time Computation of a Class of DFT Algorithm,
IEEE Transactions on Acoustics, Speech, and signal Processing, Vol. ASSP-34, No. 5,
October 1986, pp. 1301-1309.
(impact factor 1.463/1992).

M2: Mendeleyevization/Catalyst

[Milutinovic87c] Milutinovic, V.,
A Simulation Study of the Vertical-Migration Microprocessor Architecture,
IEEE Transactions on Software Engineering, Vol. SE-13, No. 12, December 1987, pp.
1265-1277.

H1: Hybridization/Symbiosis

[Milutinovic85] Milutinovic, V.,
A Microprocessor-Oriented Algorithm for Adaptive Equalization,
IEEE Transactions on Communications, Vol COM-33, No 6, June 1985, pp. 522-526.
(impact factor 1.512/2010).

H2: Hybridization/Synergy

[Milutinovic87b] Milutinovic, V., Lopez-Benitez, N., Hwang, K.,
A GaAs-Based Microprocessor Architecture for Real-Time Applications,
IEEE Transactions on Computer, Vol C-36, No 6, June 1987, pp. 714-727.
(impact factor 1.822/2010).

T1: Transdisciplinaryization/Modification

[Milutinovic86b] Milutinovic, V.,
GaAs Microprocessor Technology,
IEEE Computer, Vol 19, No. 10, October 1986 (Invited, Guest Editor's Introduction), pp.
10-15.
(impact factor 2.205/2010).

T2: Transdisciplinaryization/Mutation

[Milutinovic87a] *Milutinovic, D., Milutinovic, V., Soucek, B., The Honeycomb Architecture, IEEE Computer, Vol. 20, No. 4, April 1987 (Open Channel), pp. 81-83. (impact factor 2.205/2010).*

R1: Remodeling/Granularization

[Milutinovic88] *Milutinovic, V., A Comparison of Suboptimal Detection Algorithms Applied to the Additive Mix of Orthogonal Sinusoidal Signals, IEEE Transactions on Communications, Vol. COM-36, No. 5, May 1988, pp. 538-543.*

R2: Remodeling /Reparametrization

[Milutinovic89] *Milutinovic, V., Bettinger, M., Helbig, W., Multiplier/Shifter Design Trade-offs in a 32-bit Microprocessor, IEEE Transactions on Computers, Vol. 38, No. 6, June 1989, pp. 847-880. (impact factor 1.822/2010).*

U1: Unorthodoxization/ViewFromAbove

[Milutinovic2000] *Milutinovic, V., Cvetkovic, D., Mirkovic, J., "Genetic Search Based on Multiple Mutation Approaches," IEEE Computer, 2000. (impact factor 1.822/2010).*

U2: Unorthodoxization/ViewFromInside

[Milutinovic2001] *Milutinovic, V., Ngom, P., Stojmenovic, I., "STRIP --- A Strip Based Neural Network Growth Algorithm for Learning Multiple-Valued Functions," IEEE Transactions on Computers, 2001. (impact factor 1.822/2010).*

9. General References

Education Related References

[Milutinovic95] Milutinovic, V., "The Best Method for Presentation of Research Results," *IEEE TCCA*, September 1995.

[Milutinovic96] Milutinovic, V., 1996 "A Good Method for Presentation of Research Results," *IEEE TCCA*, March 1996.

[Milutinovic2008] Milutinovic, V., Tomazic, S., "How to Ruin the Career of a PhD Student," *IPSI Transactions*, July 2008.

[Milutinovic2012] Milutinovic, V., Korolija, N., *A Short Course for PhD Students in Science and Engineering: "How to Write Papers for JCR Journals," to be submitted, January 2012.*

Methodology Related References

[Kline86]
http://books.google.com/books?hl=en&lr=&id=-nT3zZRenrUC&oi=fnd&pg=PA275&dq=methodologies+for+research+innovation+in+science+and+engineering&ots=ZQLJm0WzF&sig=p1h6d0mjblCBcdBrdNt0CcN7m_I

[Faulkner94]
<http://sth.sagepub.com/content/19/4/425.short>

[Cummings2005]
<http://sss.sagepub.com/content/35/5/703.short>

[Libarkin2002]
https://www.msu.edu/~libarkin/Publications_files/ResMeth-v50n4p449.pdf

[Linn87]
<http://onlinelibrary.wiley.com/doi/10.1002/tea.3660240302/abstract>

[Swan97]
<http://onlinelibrary.wiley.com/doi/10.1111/1467-8551.0050/abstract>

[Perkmann2007]
<http://onlinelibrary.wiley.com/doi/10.1111/j.1468-2370.2007.00225.x/full>

[Pavon2005]
<http://www.igi-global.com/chapter/agent-oriented-methodologies/5061>

[Iivari2007]
<http://dl.acm.org/citation.cfm?id=2017331>

[Wang2003]
http://en.cnki.com.cn/Article_en/CJFDTOTAL-JSJJ200312009.htm

[Dorfler2011]

Stierand, M. & Dorfler, V., "Methods against Methods," Anabela Mesquita Technology for Creativity and Innovation: Tools, Techniques and Applications, IGI Global, Hershey, 2011, PA, 121-134.

[Dorfler2010]

Dorfler, V., "Fit for Innovation, Adam Jolly The Innovation Handbook: How to Profit from your Ideas," Intellectual Property and Market Knowledge, Kogan Page, London, 2010, 150-153.

[DorflerAckermann2012]

Dorfler, V. & Ackermann, F., "Understanding Intuition: The Case for Two Forms of Intuition, Management Learning", 2012

[DorflerBaracskai2010]

Dorfler, V., Baracskai, Z., and Velencei, J., "Understanding Creativity," Transactions on Advanced Research, 2010, 6(2): 17-25.

10. Acknowledgements