Micro-learning in Improving Professional Competences of Programmers: Pilot Studies

Anna Stolińska, Wojciech Baran, Jozef Kapusta, and Katarzyna Wójcik

1 Introduction

Programming skills are recognized by the European Commission as one of the key competencies, included both in the Digital Education Action Plan 2021–2027 (2021) and in the conclusions of the European Council formulated in December 2017 for all throughout life. The Heads of State and Government stated that programming is one of the key competences (Developing Key Competences, 2021). The ability to program is not only associated with the possibility of professional development (preparation for the profession of a programmer) but is also believed to offer the possibility of the general development of problem-solving skills in various areas of life (Martín-Ramos et al., 2017) and the growth of creativity and collaboration. The programming ability is also needed for many jobs, because currently more than 90% of professional occupations nowadays require digital competences, including programming (Coding – the 21st century skill, 2021). The rapid development of ICT and new technologies puts pressure on not only educating young people in programming but also enabling working people to acquire programming skills (Katane & Katans, 2018). The demand for programmers exists not only in Europe but also around the world. For example, on the official website of the United States government, you can read that: *Employment of software developers, quality assurance analysts, and testers is projected to grow 22 percent from 2019 to 2029,*

A. Stolińska (✉)
Department of Computer Science, College of Economics and Computer Science, Kraków, Poland
e-mail: astolinska@wsei.edu.pl

W. Baran · J. Kapusta · K. Wójcik
Pedagogical University of Krakow, Institute of Computer Science, Kraków, Poland
e-mail: wojciech.baran@up.krakow.pl; jozef.kapusta@up.krakow.pl;
katarzyna.wojcik@up.krakow.pl

E. Smyrnova-Trybulska et al. (eds.), *Microlearning*,
https://doi.org/10.1007/978-3-031-13359-6_5

much faster than the average for all occupations. These workers will be needed to respond to an increased demand for computer software (Software Developers, 2021).

The growing demand for programmers has highlighted the problems related to coding education, in particular a talent shortage as the education system is slow to react to new demands (Coding – the 21st century skill, 2021). Moreover, learning to program is perceived as difficult. There are many problems in teaching programming that researchers write about (Tsai, 2019; Shefer et al., 2018, Ouahbi et al., 2015; Basawapatna, 2016). Many students have difficulty mastering abstract programming concepts such as conditionals and loops, syntax problems of different languages, and constantly applying what they learn to new and unknown problems (Butler & Morgan, 2007). It turns out that students devote a great deal of time learning syntax and semantics while searching for solutions to problems seems to be marginalized (Andrzejewska et al., 2016). These and other problems show that teaching programming requires effort, learner involvement, and individualization in the learning process. Isong (2014) proposes a departure from traditional methods of teaching programming based on a lecture, demonstration with instruction. The researcher believes that teachers must make more use of ICT-supported learning environments. Programming teaching methods should promote the active participation and involvement of students. The answer to these problems may be, inter alia, proposing micro-learning courses. This concept is in line with the theory of Baumgartner (2013), who argues that professional knowledge is irreducible, complex, uncertain, instable, and unique. The characteristics of professional knowledge assume that we live in an inherently turbulent environment, with undefined problem situations that are "not in the book" and it is micro-learning that can provide an appropriate learning environment to support creative problem solving and inventing new things. Micro-lessons can provide knowledge quickly, inspire to create simple and effective solutions, and at the same time present content that is easily accessible, also from mobile devices (Hug, 2006). Moreover, micro-learning is closer to the already natural learning methods of young people as it is adapted to their attention span (Jaokar, 2007).

Micro-content is available through various platforms, one of the most popular is Youtube (Moghavvemi et al., 2018). It seems, however, that a very good proposal is to create generally available micro-lection courses, prepared by specialists, so that the knowledge provided is reliable and factually correct, and at the same time well thought out in terms of its structure. In our opinion, these requirements are met by the courses on the Priscilla platform, developed under the Erasmus+ Capacity Building in the Field of Higher Education project's No. 2018-1-SK01-KA203-046382 "Work-based learning in future it professionals education".

To summarize these facts, there is still no answer to the question of how professional programmers assess the effectiveness of learning programming with the use of micro-learning.

2 Background

Learning to program begins more and more in primary school (Serafini, 2011; Fatourou et al., 2018) and continues through the stages of education. A report by European Schoolnet (2018), a network of 33 European ministries of education based in Brussels, a non-profit organization that strives to innovate teaching and learning, shows that more and more ministries of education are analyzing the issues of teaching programming and looking for answers to questions how to define life-long learning in the field of programming, how to design the programming science so that it influences the development of students' skills in the twenty-first century.

However, preparation for the profession of programmer requires specialized training in vocational education at the secondary or universities level. The specific education programs and scopes vary by country, but the overall framework for programming education is very similar (Robins et al., 2003). Teaching strategies, programming languages, or supporting tools such as the integrated development environment, IDE, also differ slightly (Pears et al., 2007). In the case of programming languages, their choice is determined not only by didactic reasons, but also by the labor market. The popularity ranking dictates the trends to which schools and universities adapt. This fact was emphasized by Cass (2020), who published research on the popularity of programming languages.

Learning to program is considered difficult, and difficulties (although different) are experienced by both students and teachers – both for students and teachers. Hence, various teaching concepts arise in order to adapt them to the needs, abilities and preferences of students. Recently, there has also been a limited amount of research into adult programming education (Begel & Simon, 2008; Chilana et al., 2016; Dorn & Guzdial, 2010; Ericson et al., 2016). And yet the change of profession, retraining, is part of today's professional careers. Interesting research on programming learning by older adults was conducted in 2017. Using an online survey of 504 respondents aged 60–85, coming from 52 different countries, it was found that older adults were motivated to learn to keep their mind healthy as they got older, make up for lost opportunities in their youth, make up contact with younger family members and improve career prospects (Guo, 2017).

Programming courses are also generally considered challenging, and often have the highest dropout rates. It is generally accepted that it takes about 10 years of experience to turn a novice into a proficient programmer (Robins et al., 2003). It is also not easy to become a programmer just after learning on your own. More and more people are learning programming on their own, and a lot of people who learned independently apply for a job. But in this case, a little discipline and motivation are not enough – it is very important to know where to get knowledge and good practices from.

In learning programming, you can use various sources and teaching aids. The most popular are:

1. Tutorials and documentation – this is the first source worth consulting. By familiarizing yourself with these materials, it should be possible to quickly determine which language suits the learner best.
2. Books and e-books – it can be said that learning from books is not the most convenient form of learning, because it does not provide practical learning, but rather focuses on theory. It is inconvenient to rewrite multi-page code.
3. Programming blogs – in this case, there is a risk of acquiring bad habits or even incorrect knowledge. However, they are often a source of novelty in a given programming language.
4. Development community support – e.g. on StackOverflow.
5. Video courses are an increasingly popular method, thanks to which you can learn programming both from the theoretical and practical side. The available courses can be paid or free.
6. Bootcamps – they involve learning with a teacher (mentor) who orders tasks to be performed, helps to solve them, shows sources that can be used. Bootcamps allow you to gain extensive knowledge, the training is intensive, focused on practical knowledge. The material covers not only the basics of programming, but also the science of technology that will be useful in the future work of a programmer. Bootcamps are most often held online, although there are also those where classes are held in a lecture hall. Classes last from several months to even a year. Such classes require a lot of systematic work, you need to devote several hours a day to learning (Tu et al., 2018).

These methods can be supplemented by online micro-learning interactive courses. The method is based on getting to know theoretical knowledge and then solving short tasks that are checked by the system. The courses are adapted to various levels of advancement, they are flexible, and you can learn at your own pace, at any time convenient for you (Zhang & Ren, 2011). It should be noted, however, that there is research indicating that microlearning courses can be useful in teaching programming. Skalka i Drlik (2018) described conceptual framework of microlearning-based training for improving programming skills. Researchers have addressed the topic of teaching programming using microlearnig, but it is difficult to find references in the literature to studies that discuss the issue of improving the competencies of adults (working professionals).

3 Research Design

In order to determine to what extent IT specialists (professional programmers) acquired, acquire or would like to acquire programming skills with the use of the micro-learning type of teaching method, a survey was conducted with the use of an Internet questionnaire (CAWI). The survey questionnaire consisted of several questions and was structured in such a way that the respondents answered only those questions that were related to their situation. This non-linear questionnaire consisted of single- and multiple-answer questions, and the 7-point Likert scale was used in most of the questions about the ratings, opinions of the respondents. The link to the survey was posted on polish websites: on LinkedIn, Facebook and sent to 4 IT companies. Data was collected for 4 days. 50 people took part in the study, of which 35 were men (70%), 12 women (24%), and 3 people (6%) refused to provide information about their gender.

In the study participated programmers that use various languages, as shown in Fig. 1. The most popular language turned out to be JavaScript, in which 27 respondents (54%) program, as well as PHP and Java (17 [34%] and 16 [32%] participants, respectively. research).

Basic statistics on age and seniority (the measure used was a year) are presented in Table 1.

Most of the respondents are young people (Mean = 26.8 years, SD = 4.7), with a short work experience (Mean = 3.8 years, SD = 4.02). 11 people (22%) are employed as Full Stack Developer, 9 (18%) – Software Engineer, 7 (14%) – Back-end Developer, 5 people (10%) – Front-end Developer. 23 people (46%) described their level of professional experience as a medium developer / regular developer, 19 people (38%) as a junior developer, 7 people (14%) – a senior developer. 26 respondents (52%) work in a large company employing over 250 people, 13 (26%) in a small company (10–50 employees), 8 (16%) in a medium company (51–250 employees).

The main research problem concerned the way in which people working as programmers improve their coding skills. Finding an answer to this question required the formulation of detailed research questions, among which they were included:

1. How did IT specialists acquire the programming skills (competences)?
2. How many programmers declare improving their programming competences during their professional career?
3. What forms of training do professional programmers use?

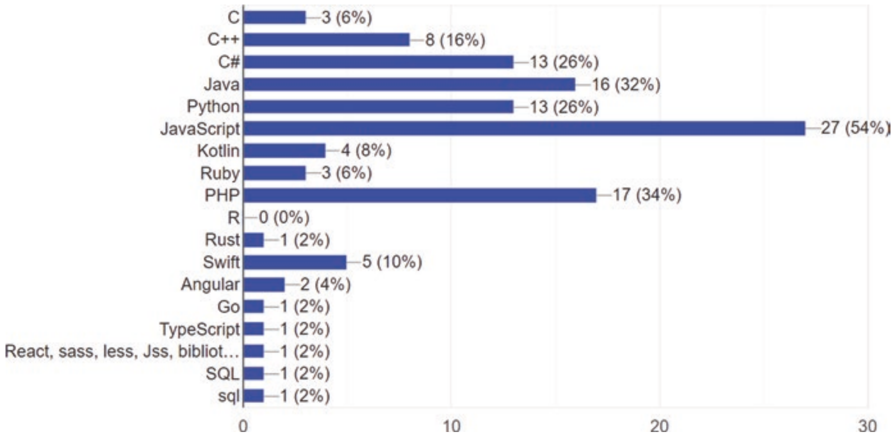

Fig. 1 Programming languages dominating in the respondents' current tasks/professional projects

Table 1 Age and work experience in the profession of a programmer

Age		Number of years of professional experience as a programmer	
Mean	26.79	Mean	3.83
Standard deviation	4.66	Standard deviation	4.02
Median	25.00	Median	3.00
Dominant	25.00	Dominant	4.00
Kurtosis	4.92	Kurtosis	10.62
Skewness	2.03	Skewness	2.84
Range	23.00	Range	23.00
Minimum	22.00	Minimum	0.00
Maximum	45.00	Maximum	23.00
Trust level (95.0%)	1.32	Trust level (95.0%)	1.14

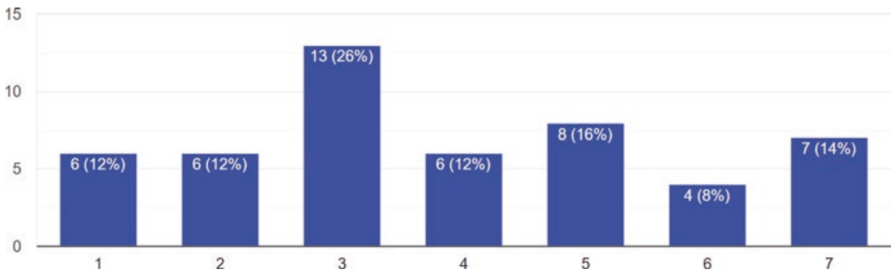

Fig. 2 Assessment of the preparation obtained during studies to work in the profession of a programmer

1. How many developers are using a learning method called micro-learning?
2. To what extent is it used by micro-learning programmers as a method of learning / improving competences?

4. What is the interest of programmers in using micro-learning in improving their professional competences?

4 Results

4.1 *How Did IT Specialists Acquire the Programming Skills (Competences)?*

The vast majority of the respondents, ie 46 people (92%) graduated in computer science, the others in related fields of study, only two people indicated that they did not study the field of computer science. The respondents indicated on a scale from 1 to 7, where 1 means very poorly to 7 (very good), how they assess the preparation obtained during their studies to work as a programmer. The results are shown in Fig. 2.

The respondents assessed their preparation for work in the profession of programmer at the level M = 3.5 (SD = 1.9).

4.2 How Many Programmers Declare Improving their Programming Competences During their Professional Career?

As many as 98% of the respondents (49 people) declared improving their programming skills during their professional career.

4.3 What Forms of Training Do Professional Programmers Use?

The respondents supplemented their knowledge and skills in programming in various ways. 44 people (88%) learned using the content (lectures) posted on various websites, e.g. Youtube, 39 (78%) from books and magazines, 22 (44%) participated in stationary training courses, 8 people (16%) attended bootcamps. Currently, while working, most of them declare that they independently find knowledge / skills while implementing projects and solving problems (41 people, 83.7%), many still use YouTube (39 people, 79.6%) or learn from books, trade magazines (22, i.e. 44.9% of respondents).

4.3.1 How Many Developers Are Using a Learning Method Called Micro-learning?

Most of the respondents (37 people, 74%) declared that they use micro-learning. The frequency of using this teaching method varies – 8 (21.6%) people indicated the option "very often, almost every day", 12 people (32.4%) – "often, several times a week", the option "sometimes, what at most a few times a month" was indicated by 10 people (27%), rarely (several times a year) – 7 people (18.9%). People who declared that they do not use micro-learning, as the reason for this state of affairs stated that they have not experienced this type of course (11 people), 2 people said that they do not like learning this way – they prefer longer materials, exactly describing a given issue, problem, another two people said that programming is coding – you can't learn something just by reading or watching.

4.3.2 To What Extent Is It Used by Micro-learning Programmers as a Method of Learning/Improving Competences?

The vast majority of respondents who declared to learn using the micro-learning method, at the same time indicated that they learned this method of programming (33 people, 89.2%). Below are the respondents' opinions on micro-learning in programming teaching, with each rating related to a given statement being formulated on a scale from 1 to 7, where 1 meant: completely disagree and 7: completely agree (Table 2).

4.4 What Is the Interest of Programmers in Using Micro-learning in Improving their Professional Competences?

The respondents rated their interest in micro-learning courses on a scale from 1 to 7, where 1 meant no interest, 7 – high interest. The average interest in micro-learning courses was M = 5.5 (SD = 1.87). A value below 4 was indicated by 8 people, of which 3 declared a total lack of interest in micro-learning in the field of programming.

Table 2 Assessment of micro-learning in learning programming

Learning with the use of micro-learning is:	Efficient (you can learn a lot)	Convenient (you can study anywhere. anytime)	Fast (small bits of knowledge can be mastered quickly)	Flexible (you can only choose what you need at the moment)	Common (many people I know learn this way)
Average	5.51	5.78	5.68	5.92	4.68
Standard deviation	1.43	1.78	1.67	1.46	1.94
Median	6	7	6	7	5
Dominant	7	7	7	7	7
Kurtosis	−0.43	0.02	1.01	0.75	−1.28
Skewness	−0.64	−1.21	−1.31	−1.32	−0.23
Range	5	5	6	5	6
Minimum	2	2	1	2	1
Maximum	7	7	7	7	7
Trust level (95.0%)	0.48	0.59	0.56	0.49	0.65

5 Discussion

Most of the time, programmers who were at the beginning of their professional career took part in the study. It is not surprising, therefore, that research indicates a high activity of IT specialists in the field of improving programming skills. The use of various forms and methods of training has been confirmed, while the frequency of using textbooks and trade magazines seems to be quite high compared to participation in bootcamps or stationary courses.

6 Conclusion

The research confirmed that the set of micro-learning courses prepared as part of the "Work-based learning in future it professionals education" project can and should also be made available to professional programmers who want to improve their skills in other languages, or to improve those already possessed. It also seems advisable to build a community of programmers around the Priscilla platform, who would enrich its content with tasks, mini-problems that would strongly relate to projects implemented at work.

Pilot studies also allow for the formulation of certain conclusions that should be taken into account during subsequent, already relevant studies. You need to diagnose what specific knowledge programmers are looking for and from what specific materials (where are they made available? Are they peer-reviewed?) They learn as part of professional development.

References

Andrzejewska, M., Stolińska, A., Błasiak, W., Peczkowski, P., Rosiek, R., Rożek, B., Sajka, M., & Wcisło, D. (2016). Eye-tracking verification of the strategy used to analyse algorithms expressed in a flowchart and pseudocode. *Interactive Learning Environments, 24*(8), 1981–1995. https://doi.org/10.1080/10494820.2015.1073746

Basawapatna, A. (2016). Alexander meets Michotte: A simulation tool based on pattern programming and phenomenology. *Journal of Educational Technology & Society, 19*(1), 277–291.

Baumgartner, P. (2013). Educational dimensions of microlearning – Towards a taxonomy for microlearning. In P. A. Bruck & M. Sedlaczek (Eds.), *Designing microlearning experiences – Building up knowledge in organisations and companies*. Innsbruck University Press.

Begel, A., & Simon B. (2008). Struggles of new college graduates in their first software development job. In *Proceedings of the 39th SIGCSE technical symposium on computer science education (SIGCSE '08)* (pp. 226–230). New York: ACM. https://doi.org/10.1145/1352135.1352218.

Butler, M., & Morgan, M. (2007). Learning challenges faced by novice programming students studying high level and low feedback concepts. In *Proceedings of ASCILITE – Australian Society for Computers in Learning in Tertiary Education, annual conference 2007* (pp. 99–107). Retrieved 15 July 2021 from https://www.learntechlib.org/p/46043/

Cass, S. (2020). The top programming languages: Our latest rankings put Python on top-again – [careers]. *IEEE Spectrum, 57*(8), 22–22. https://doi.org/10.1109/mspec.2020.9150550

Chilana, P. K., Singh, R., & Guo, P. J. (2016). Understanding conversational programmers: A perspective from the software industry. In *Proceedings of the 2016 CHI conference on human factors in computing systems (CHI '16)* (pp. 1462–1472). New York: ACM. https://doi.org/10.1145/2858036.2858323

Coding – The 21st century skill. European Commission. Retrieved 1 July 2021 from https://ec.europa.eu/digital-single-market/en/coding-21st-century-skill

Developing key competences for all throughout life. European Commission. Retrieved 1 July 2021 from https://ec.europa.eu/education/sites/default/files/document-library-docs/factsheet-key-competences-lifelong-learning_en.pdf

Digital education action plan. European Commission. Retrieved 1 July 2021 from https://ec.europa.eu/education/education-in-the-eu/digital-education-action-plan_en

Dorn, B., & Guzdial, M. (2010). Learning on the job: Characterizing the programming knowledge and learning strategies of web designers. In *Proceedings of the SIGCHI conference on human factors in computing systems (CHI'10)* (pp. 703–712). New York: ACM. https://doi.org/10.1145/1753326.1753430

Ericson, B. J., Rogers, K., Parker, M., Morrison, B., & Guzdial M. (2016). Identifying design principles for CS teacher Ebooks through design-based research. In *Proceedings of the 2016 ACM conference on international computing education research (ICER'16)* (pp. 191–200). New York: ACM. https://doi.org/10.1145/2960310.2960335

European Schoolnet's 2018 annual report. European Schoolnet. Retrieved 2 June 2021 from http://www.eun.org/documents/411753/817341/activity-report_2018_online_FINAL.pdf

Fatourou, E., Zygouris, N. C., Loukopoulos, T., & Stamoulis, G. I. (2018). Teaching concurrent programming concepts using scratch in primary school: Methodology and evaluation. *International Journal of Engineering Pedagogy, 8*(4), 89–105. https://doi.org/10.3991/ijep.v8i4.8216

Guo, P. J. (2017). Older adults learning computer programming: Motivations, frustrations, and design opportunities. In *Proceedings of the 2017 CHI conference on human factors in computing systems*, May 2017 (pp. 7070–7083). https://doi.org/10.1145/3025453.3025945

Hug, T. (2006). Microlearning: A new pedagogical challenge (introductory note). In T. Hug, M. Lindner, & P. A. Bruck (Eds.), *Microlearning: Emerging concepts, practices and technologies after E-learning, Proceedings of microlearning conference 2005: Learning & Working in New Media*. Innsbruck University Press.

Isong, B. (2014). A methodology for teaching computer programming: First year students' perspective. *International Journal of Modern Education and Computer Science, 6*, 15–21. https://doi.org/10.5815/IJMECS.2014.09.03

Jaokar, A. (2007). Mobile Web 2.0, micro-learning, intertwingularity, and mobile widgets. *Educational Technology, 47*(6), 43–45.

Katane, I., & Katans, E. (2018). Environmental contexts of programmer's professional self-development through learning: Ecological and synergetic approach. In *Proceedings of the 17th International Scientific Conference "Engineering for Rural Development"*, Jelgava, Latvia, May 23–25 2018. https://doi.org/10.22616/ERDEV2018.17.N057

Martín-Ramos, P., Lopes, M. J., da Silva, M. M. L., Gomes, P. E. B., da Silva, P. S. P., Domingues, J. P. P., & Silva, M. R. (2017). First exposure to Arduino through peer-coaching: Impact on students' attitudes towards programming. *Computers in Human Behavior, 76*, 51–58. https://doi.org/10.1016/j.chb.2017.07.007

Moghavvemi, S., Sulaiman, A., Jaafar, N. I., & Kasem, N. (2018). Social media as a complementary learning tool for teaching and learning: The case of Youtube. *The International Journal of Management Education, 16*(1), 37–42.

Ouahbi, I., Kaddari, F., Darhmaoui, H., Elachqar, A., & Lahmine, S. (2015). Learning basic programming concepts by creating games with Scratch programming environment. *Procedia – Social and Behavioral Sciences, 191*, 1479–1482. https://doi.org/10.1016/j.sbspro.2015.04.224

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M., & Paterson, J. (2007). A survey of literature on the teaching of introductory programming. *ACM SIGCSE Bulletin, 39*(4), 204–223.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education, 13*, 137–172. https://doi.org/10.1076/csed.13.2.137.14200

Serafini, G. (2011). Teaching programming at primary schools: Visions, experiences, and long-term research prospects. In *Proceedings of the 5th international conference on informatics in schools: Situation, evolution and perspectives*. Berlin, Heidelberg: Springer-Verlag.

Shefer, O. P., Nosova, L. S., & Lebedeva, T. N. (2018). A modern methodology for teaching programming at a university. *Scientific and Technical Information Processing, 45*(2), 81–86. https://doi.org/10.3103/S0147688218020077

Skalka, J., & Drlik, M. (2018). Conceptual framework of microlearning-based training mobile application for improving programming skills. In M. Auer & T. Tsiatsos (Eds.), *Interactive Mobile communication technologies and learning. IMCL 2017* (Advances in intelligent systems and computing) (Vol. 725, pp. 213–224). https://doi.org/10.1007/978-3-319-75175-7_22

Software developers, quality assurance analysts, and testers, U.S. Bureau of Labor Statistics. Retrieved 1 July 2021 from https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm

Tsai, C.-Y. (2019). Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. *Computers in Human Behavior, 95*, 224–232. https://doi.org/10.1016/j.chb.2018.11.038

Tu, Y-C., Dobbie, G., Warren, I., Meads, A., & Grout, C. (2018). An experience report on a boot-camp style programming course. In *Proceedings of the 49th ACM technical symposium on computer science education*, February 2018 (pp. 509–514). https://doi.org/10.1145/3159450.3159541.

Zhang, X., & Ren, L. (2011). Design for application of micro learning to informal training in enterprise. In *2nd international conference on artificial intelligence, management science and electronic commerce (AIMSEC)* (pp. 2024–2027). https://doi.org/10.1109/AIMSEC.2011.6011235.