Applied Soft Computing 25 (2014) 86–96

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Genetic programming for smart phone personalisation

Philip Valencia, Aiden Haak, Alban Cotillon, Raja Jurdak*

Autonomous Systems Laboratory, CSIRO Computational Informatics, Brisbane, Australia

ARTICLE INFO

Article history:
Received 20 August 2012
Received in revised form 8 August 2014
Accepted 8 August 2014
Available online 16 September 2014

Keywords:
Genetic Programming
Island Model
Personalization
Smart phone
Online evolutionary

ABSTRACT

Personalisation in smart phones requires adaptability to dynamic context based on user mobility, application usage and sensor inputs. Current personalisation approaches, which rely on static logic that is developed a priori, do not provide sufficient adaptability to dynamic and unexpected context. This paper proposes genetic programming (GP), which can evolve program logic in realtime, as an online learning method to deal with the highly dynamic context in smart phone personalisation. We introduce the concept of collaborative smart phone personalisation through the GP Island Model, in order to exploit shared context among co-located phone users and reduce convergence time. We implement these concepts on real smartphones to demonstrate the capability of personalisation through GP and to explore the benefits of the Island Model. Our empirical evaluations on two example applications confirm that the Island Model can reduce convergence time by up to two-thirds over standalone GP personalisation.

1. Introduction

Smartphones have experienced exponential growth in recent years. These phones embed a growing diversity of sensors, such as gyroscopes, accelerometers, Global Positioning Systems (GPS), and cameras, with broad applicability in areas such as urban sensing or environmental monitoring. Coupled with phone users' high mobility and diverse profiles [2], this sensory richness has fuelled complex applications with composite logic, including features such as location-based and usage-based services. The increased application complexity involves significant challenges in personalising smart phone applications so that they can adapt to new or unexpected context.

Smartphone personalisation either occurs centrally at a server or locally at the phone. Centralised approaches track user activity to customise content delivery or application behaviour. They can easily misrepresent user preferences due to spurious activity, and they involve privacy concerns as users need to share their data with the content providers. Most standalone smartphone algorithms, aiming at either data-centric [5] or user-centric personalisation [3], are based on static or rule-based approaches. However, personalisation increasingly depends on contextual information and user inputs [4], which are both subject to dynamic changes arising from mobility and user preferences. The problem of personalisation of smart phones is therefore multidimensional and requires an approach that can not only adapt parameters in response to changes in user preferences and context, but also adapt the program logic to optimise for unpredictable changes in context.

Online learning is well-suited for smart phone personalisation. In order to provide maximum versatility to support the creativity and unpredictability of smart phone users, we need an online learning approach that allows for the adaptation of program logic, and not just parameters within fixed program logic. Additionally, smart phone users are often co-located with several other users that share their context, providing opportunities for collaborative personalisation based on this shared context. The most suitable online learning strategy should provide the syntactic richness to evolve logic on a single smartphone, and to support the sharing of functional logical blocks among co-located phones according to the building block hypothesis [34].

Most online learning methods, such as reinforcement learning [6] and neural networks [7], are concerned with parameter optimisation only. While these methods could be run within an evolutionary framework, any sharing of genetic sequences across phones would not necessarily map to functional logic blocks, which could slow down convergence. Genetic Programming (GP), on the other hand, is amenable to this scenario, as it evolves both logic and parameters simultaneously, providing it with the syntactic richness and flexibility that is comparable to offline human software development. Because GP evolves functional logic blocks, sharing logic across multiple phones is both more meaningful and more conducive to quicker convergence.

This paper proposes GP for smart phone personalisation. We first show empirically that GP can support smart phone

* Corresponding author.
E-mail address: rjurdak@ieee.org (R. Jurdak).

http://dx.doi.org/10.1016/j.asoc.2014.08.058
1568-4946/

personalisation through our software framework called the Android Genetic Programming Framework (AGP). In order to expedite convergence towards high performing applications, we propose collaborative personalisation of smartphones through the GP Island Model to exploit shared context between co-located phones. The main research question we aim to answer is: "To what extent does collaborative smartphone personalisation improve convergence time?". To address this question, we extend AGP to support the island model and run extensive experiments with 2 Android phones for a comparative evaluation of the benefits of sharing logic among smartphones against *stand-alone* personalisation, exploring 6 scenarios with different migration rates and intensities. Our results show that the Island Model consistently outperforms the standalone GP for online personaliation, and somewhat surprisingly, that injecting random programs into the subpopulations can be beneficial for the more complex application of energy-efficient localisation.

The contributions of this paper are:

- Introduction, motivation, and demonstration of genetic programming for smart phone personalisation
- Proposal of collaborative smart phone personalisation through the GP Island Model for faster convergence and exploitation of shared context
- Empirical evaluation of both standalone and collaborative personalisation through two case study applications, which confirm the benefits of collaborative personalisation

The remainder of the paper is organised as follows. Section 2 discusses related work in the literature. Section 3 defines the problem for smart phone personalisation, motivates genetic programming to address this problem, and proposes collaborative GP for faster personalisation. Section 4 briefly introduces the AGP framework and our extension to support the Island Model. Section 5 demonstrates online personalisation through GP, while Section 6 evaluates the benefits of collaborative personalisation through the Island Model. Section 7 discusses the results and concludes the paper.

2. Related work

Most online learning approaches, including neural networks, adaptive systems, and reinforcement learning, use specialised structures for representation within the learning process [8]. Because of their reliance on specialised structures, these approaches are amenable for online learning situations where the overall program logic is well-defined while individual parameters within this program structure need to be optimised. While these approaches have been used in an evolutionary context (for e.g. in [31]), their representation of genetic material is not necessarily aligned to functional logic blocks, which makes them less amenable to logic sharing among islands. In contrast, genetic programming uses program logic representation in the learning process, supporting high versatility for entirely new actions in response to unpredictable stimuli. Through its program tree representation, GP can isolate functional logic blocks for sharing across islands to leverage shared context for faster convergence.

Several previous GP works in have focused on architectural issues. Whereas some solutions provide generic frameworks for evolutionary computation problems [10,11,9], others propose application-specific solutions. Ismail et al. [12] describe a GP framework for extracting a mathematic formula needed for fingerprint matching, whereas, the authors in [13] focus on a genetic programming framework for content-based image retrieval. In [16], Lacerda et al. introduce a framework for associating ads with web pages based on GP. Valencia et al. [14] study genetic programming for Wireless Sensor Networks and propose the In Situ Distributed Genetic Programming (IDGP) framework. DGPF [17] brings utilities for Master/Slave, Peer-to-Peer, and P2P/MS hybrid distributed search execution. P-Cage [25] introduces and evaluates a complete framework for the execution of genetic programs in a P2P environment. It shows the relevance of using P2P networks scalability to counteract computation limitations.

Design patterns describe the interaction between groups of classes or objects. They concentrate on specific concerns for implementing source code to support program organisation. When they are well integrated into a framework, they ensure the goals of extensibility and reuse. Lenaerts and Manderick [15] discuss the construction of an object-oriented Genetic Programming framework using design patterns to increase flexibility and reusability. McPhee et al. [10] extend the latter to Evolutionary Computation (EC). As the solution search space for a problem becomes wider, it leads to a more abstract set of classes. Based on those works, Ventura et al. introduced JCLEC [9], a Java Framework for evolutionary computation. They present a layered architecture and provide a GUI for EC. This paper similarly uses Java for a genetic programming framework, albeit for a more resource constrained smart phone platform.

Smartphone personalisation research has mainly focused on rule-based approaches. Korpipaa et al. [18] introduced a first framework for user customisation using context changes as triggers. Their prototype enables the end-user to set up actions on context events such as GUI interactions, RFID tag informations, accelerometer peaks and other data from embedded sensors. Since this first try on Nokia N73 smartphone, similar applications have been published on both Android and Apple applications stores. Onx [19] and Launch Center Pro [20] let users build rules for automating various tasks. These mechanisms still require explicit user involvement in personalisation, which limits their utility to more technology savvy users.

More recently, Interactive Differential Evolution (IDE) has been applied on smart phones [1] as a method to achieve quick image enhancement of photos taken on mobile phones. As with most Interactive Evolutionary Computation (IEC) implementations, IDE encodes parameters of a fixed-logic solution as a vector and optimises these parameters over time, unlike GP which also optimises the logic.

IEC methods include interactive evolution strategy [27], interactive genetic algorithm [30], interactive genetic programming [29], and human-based genetic algorithm [28]. An interactive genetic algorithm (IGA) is defined as a genetic algorithm that uses human evaluation.

The community-based earthquake detection technique, proposed by Faulkner et al. [32], highlighted the need to consider distributed context, using accelerometer data from multiple smartphones to detect earthquakes. The distributed nature of the data allowed the detection and filtering out of false positives and negatives due to spurious sensor data. This highlights the importance of distributed context for the collaborative learning method in this paper.

Another instance of related work is activity recognition using accelerometers data [33]. Interestingly, Weiss found that using recognition models tailored specifically to a user outperformed an impersonal model which used data gathered from multiple users. The personal model also outperformed a hybrid model using a combination of a model based on data from both a specific user and a model based on data gathered from multiple users. However, it must be noted that these models were all developed offline and did not adapt to their users. The work in this paper postulates and shows that collaborative learning can be useful for adapting to shared context for some applications, while other problems such