An Efficiency Optimization Method for Mobile Application with Reconfigurable Software

Yonghua Zhu, Congqi Xia

Computing Center of Shanghai University, Shanghai University, Bldg D, ShangDa Road, Shanghai, China
zyh@shu.edu.cn

Abstract. This paper discusses the efficiency optimization for mobile application. By adopting reconfigurable software architecture, a dynamic reschedule method is proposed. Modeling the overheads of single operation, the uncertainty caused by control structure is represented with experienced data and branch prediction. Several criteria are proposed considering different situation.

Keywords: Mobile Application, Efficiency Optimization, Reconfigurable Software

Introduction

With the rapid development of mobile communication network, better quality of service is needed. At present, the size of the mobile communication network is substantial; improving the quality of network services in the future will mainly depend on the detailed system of network optimization project. However, due to the nature of the mobile device itself, it is necessary to consider the computing and storage capacity of the equipment when developing application on mobile device. In general, mobile devices have only limited capacity of RAM, clocked at a lower CPU and the limited capacity of the non-volatile memory. More importantly, mobile equipment has only limited battery capacity. Now there is a trend that the high frequency CPU and large memory is used in mobile devices, which makes the electricity become a constraint and an important factor.

And it has become popular that having customized mobile application. These customized applications make it possible and convenient that one can operate in a variety of business through one's mobile terminal. However, in practical applications, due to the limitations of the environment of the mobile network and mobile device computing power, data-intensive operations response time becomes very long and unbearable, which make the user experience terrible. However, data-intensive operations may be on their core business needs and is commonly used functions.

In order to solve the above problems that occur, the use of reconfigurable software framework to optimize the efficiency of the large amount of data applications for mobile terminals on the C/S structure is proposed.

Reconfigurable Software

Performance and versatility are two related issues gathering increasing attention within the embedded device co-design community. Reconfigurable computing is considered by many as a viable solution to address these future mobile device demands [1].
Mobile device management needs a server architecture which is networked, modular and flexible according to [2]. Their architecture is built using Enterprise Java Bean(EJB) technology and uses a system management API constructed with XML. Their approach offers services such as application download on demand and on-the-fly update of hardware acceleration components.
An agent-based architectural framework supporting networked reconfigurable mobile devices has been explored in our work in [1,3]. The overall aim is to enable a mobile device to dynamically retrieve an optimal, automatically partitioned and individualized reconfigurable hardware-software based application solution from a provisioning server.
There are many other frameworks that can be learned from [4-7]. In paper [4], a programming framework, based on a XML description of a set of available reconfiguration mechanisms is used to provide programmers with a development kit to write programs able to automatically reconfigure selected functionalities every time a hardware block involved into the computation is detected as faulty.

Optimization Method

This paper proposed a method using reconfigurable software architecture in order to shift the logic process according to the scheme that work best in current situation and environment. Modeling and making criteria is the key fracture of this method.

Modeling

Independent logical unit

In this paper, the independent logical unit (ILU) is proposed to express atomic tasks that can migration independently. ILU has several properties as follows:

1) Indivisibility. ILU itself cannot be split into a number of ILUs, which means the internal logic in an ILU cannot be interrupted. Otherwise, the whole program may collapse after the dynamic migration.
2) Context-free. ILU can be called in different places and the required data should be transferred as input parameters to ensure correct operation.
3) Every ILU can specify an ILU to be called after its process, but it is the system's responsibility to call the next ILU, because the next ILU may not be at the same location with the previous one.

However, not all of the ILU can be dynamically allocated to both server or client in the run-time, because each ILU may require certain physical resources situating the server side or the mobile device side, such as: ILUs that include the database operations can be implemented only on the server side, because such ILUs allocated to the mobile device side will result to a "database not find" exception; otherwise, ILUs to display specific data to the screen must be executed in the mobile device side.

Based on the above properties, a request in a C/S architecture system can be modeled as a string composed by a number of ILUs shown in Fig.2.Function CM and CT are defined in order to consider the cost of computation for each ILU. CM(ILU) stands for the computation overhead of ILU; and CT(ILU) stands for the process time of ILU. Parameter CA is introduced to model the communication cost between two ILU5, so it is possible to considering the communication cost of each different allocation strategy.

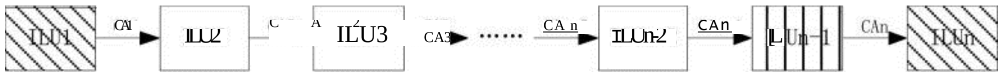

Fig. 1. A linear ILE Sequence Examples

In Fig.1, the diagonal ILUs must be running in the mobile side, the vertical striped ILUs must run on the server side and the other ILU5 from ILU2 to ILUn-2 having no requirements of physical resources which can be dispatched to both server side and client side. The load balancing can be executed base on the CAs and the CMs and CTs of every ILU.

Division of control structure of ILU

No matter express program logic in what way will encounter the branch structure and loop structure as well. These two control structures will lead uncertainty when schedule ILUs in reconfigurable software systems. In order to decide the allocation strategy before process a request, the cost of these two control structures must be estimated first.

Branch control

Due to the different weights of each branch, the ILU sequences that have branches may have different final weights. In most cases, it is not possible to determine how the process will be when receiving a request.

For such a situation, this paper presents several solutions as follows:

1) Branch Prediction. This solution is similar to the idea of branch prediction used in processors. The processor trying to predict the instruction jump using the statistics of the previously jump results. Suppose that the previous executions of a branch are all jumped, and then the next execution of this branch is much more likely to result to a jump. Before determining the next instruction, the processor will predict the next instruction into the pipeline based on the previous statistics. If the prediction is wrong, the processor will clear the pipeline and ignore all operations based on this incorrect prediction, which increases the time of computation; but, if the prediction is correct, it will significantly shorten the computation time. Generally, previous statistics leads to correct results, so that the computing speed can be substantial increase in this way. This paper takes the same idea, statistics the direction of each branch to determine the weights of every request.

2) Weight Estimate. In order to prevent the rapid decrease of the efficiency when running into wrong prediction, weight estimate is introduced in this paper. The weights for each possible path should be calculate first, then the weighted average should be calculate based on the possibility of each path to conclusion a better scheduling strategy in order to circumvent the extreme cases.

Loop control

Operations in a loop control always process many times, so that a scheduling point within the cycle may lead to a proliferation of computation. In this paper, the whole loop body is treated as a special ILU, the calculation cost CM is estimated based on the number of executions.

Factors in Decide the scheduling strategies

The previous section has modeled the reconfigurable program requests. It is easy to get the weights for each scheduling strategies. But a scheduling strategy cannot always be the best on in different environments. This section will focus on discussing the influential factors in different circumstances to make different scheduling strategies.

Response time

The communication cost can be ignored when using local network or the Internet, because it is generally fixed or at a low level. As the mainly objective turns to optimize the response time, the following formula can be established:

$$TiMe_{R„p„„} = \overset{n}{\underset{i=,}{CT(ILE,)}} + \mathrm{LcoT}(cA_{.},) \qquad (1)$$

The goal of the scheduling strategies is to achieve min($Time_{Re\prime onp\prime e}$) , which means the shortest response time.

Communicate Cost

As the mobile devices may be running in the commercial mobile network in which communication cost a lot of time and money, pursuit to reduce the response time that cause an increase on communicate cost will not be welcome by the users. In this case, the communicate cost turns to be the major optimization objective.

Communicate cost of a single request can be expressed by the following formula:

$$Cost_{com} = 1CoC(CA_i) \qquad \textbf{(2)}$$

In this case, the goal of the scheduling strategy is to obtain $min(Cosc.)$

Comprehensive consideration

In reality, too much communication will cause a lot of server pressure, while long response time will affect the user experience. Thus, in most cases, a balance is needed between the two.

$$C = aTimeRe_{sponse} + fiCost_{Com} \qquad \textbf{(3)}$$

In this case, the goal of the scheduling strategy is to obtain ^{Cmm}

As the values of the weighting coefficients a and will directly affect the scheduling strategy, a set of suitable weight coefficients can adjust both the server pressure and the user experience to an appropriate level.

Pseudo-code of proposed method

```
1. Take measurement of the communication environment
2. Decide the criteria suitable
3. Compute C for each operation
4. Launcher the operations when needed
5. Return to 1 after certain period of time or when
```

```
environment changed
```

Summary

This paper proposed a method to improve performance of mobile application in different situation. When modeling the overhead, certain prediction technique is used while different prediction technique may leads various schedule scheme. And coefficients of response time and communication amount as well. Further experiment need to be processed to explore better scheme in order to improve the performance of the method.

Reference

1. Timothy 0' Sullivan, Richard Studdert. Agent Technology and Reconfigurable Computing for Mobile Devices. 2005 ACM Symposium on Applied Computing, 2005
2. Nitsch. C., Lara. C., Kebschull. U. A Novel Design Technology for Next Generation Ubiquitous Computing Architectures. In Reconfigurable Architectures Workshop (RAW-03), 2003
3. 0' Sullivan, T., Studdert. R. Mobile Agent Technology and Networked Reconfigurable Embedded Devices. In Proceedings of International Conference on Pervasive Computing and Communications, 2004
4. Di Carlo, S. Prinetto, P. Scionti. A FPGA-Based Reconfigurable Software Architecture for Highly Dependable Systems[J] Control & Comput. Eng. Dept., Politec. di Torino, Torino, Italy 2009. Nov. 2009
5. Mast. A.W. Reconfigurable Software Defined Payload architecture that reduces cost and risk for various missions[J] Gov. Commun. Syst. Div., Harris Corp., Melbourne, FL, USA 2011
6. Congqi Xia, Yonghua Zhu. An Efficiency Optimization Strategy for Huge-Scale Data Handling.[J] Recent Advaces in Computer Science and Information Engineering, 2012(Vol.125), pp:393-398
7. Ing-Yi Chen, Chao-Chi Huang. A Reconfigurable Software Distribution Framework for Smart Living Environments[J] Nat. Taipei Univ. of Technol., Taipei 2007