

Experience Transfer for the Configuration Tuning in Large Scale Computing Systems

Haifeng Chen⁽¹⁾ Wenxuan Zhang⁽²⁾ Guofei Jiang⁽¹⁾

⁽¹⁾ NEC Laboratories America, Inc.
{haifeng, gfj}@nec-labs.com

⁽²⁾ CS Department, Rutgers University
wzhang@cs.rutgers.edu

Categories and Subject Descriptors

K.6.4 [Management of Computing and Information Systems]: System Management; G.1.6 [Numerical Analysis]: Optimization

General Terms

Algorithms, Management, Performance

Keywords

Distributed systems, configuration tuning, knowledge acquisition, knowledge reuse

1. MOTIVATION

Recent years have witnessed the emergence of autonomic management tools in large scale information systems. They utilize the knowledge learned from system experts or historical data to automate the process of management tasks. However, current autonomic solutions only focus on the knowledge discovery and modeling to benefit the management in the *same* system. It is sometimes also important to utilize the knowledge of one system to facilitate the management of other systems. For example, a lot of special kinds of systems, such as the online banking systems, usually run on similar platforms, i.e., the J2EE based infrastructure, to support applications with similar business logics. If we can learn the behavior of one system and transfer the learned knowledge to other similar systems, it is not necessary to spend the same amount of efforts and time again in modeling the new system. Furthermore, current systems evolve frequently due to the hardware upgrades, software version changes, topology changes, and so on. In those situations, the previous learned model for system management may not be valid or optimal anymore after the system undergoes changes. It is not practical to relearn the system from scratch because it needs a lot of extra time and data. If we can discover useful experiences from the management of previous systems and reuse them in the current system, the system can become more robust and adaptive to the environmental changes.

To this end, this paper proposes a new strategy, called the ‘experience transfer’, to deal with the utilization of knowledge learned from system S_0 to benefit the management of another similar system S_1 . Such transfer process consists of three main components: (1) discover and represent the experiences that can be transferrable between two systems; (2) extract the experiences during the modeling process in the original system S_0 ; (3) embed learned experiences into the management of the new system S_1 . We use the

system configuration tuning as a case application to demonstrate the process of experience transfer as well as the benefits introduced by the transfer. The configuration tuning is an important task for computing systems since an appropriate configuration setting can lead the system to the best quality of services (QoS) such as short response time, high throughput, and fairness among users. The current approaches [5][4] treat the system as a black-box and apply the sampling based optimization to identify the optimal configuration. It starts with selecting a number of samples, each of which represents a specific system configuration setting. After the configuration samples have been evaluated for the system performance, the algorithm makes an inference about the location of optimal configuration in the search space. Such inference provides feedbacks to the generation of next round sample populations. With such iterative sample generation and inference makings, the search process will eventually discover the optimal configuration of the system. However, the sampling based configuration search is a very time-consuming process. It requires tens of minutes to evaluate one *single* configuration sample because we have to try various workload situations during the evaluation in order to obtain reliable performance data. Since a common configuration search usually involves the evaluations of at least hundreds of samples, it would take several days or even weeks to complete the tuning task. Due to the expensiveness of configuration tuning, it is highly desired if we can use the knowledge learned from the tuning process in other (or previous) systems to speedup the tuning in the current system.

2. CONTRIBUTIONS

In the configuration tuning, we observe that the knowledge about dependencies between different configuration parameters plays an important role to discover the optimal configuration setting. It can help us to avoid a lot of unnecessary sample evaluations and in consequence accomplish the configuration search more quickly. For instance, if the increase of a configuration parameter c_A always introduces the increase of another parameter c_B as well in order to improve the system performance, we do not have to generate samples with high c_A and low c_B values because such value pairs will not improve the system performance. Furthermore, we observe that most of the dependencies between configuration parameters are usually unchanged across similar systems because those systems are built on same infrastructures to support applications with similar business logics. Therefore, if we can extract the parameter dependencies during the configuration tuning in S_0 , such knowledge can be utilized to speedup the tuning process in S_1 .

In light of the above observations, this paper regards the dependencies between configuration parameters as valuable experiences and uses the Bayesian network [2], a well-known technique in machine learning, to represent those experiences due to its ca-

pability in modeling the parameter dependencies. A new configuration tuning algorithm based on the Bayesian network construction and sampling is presented in Figure 1. We use BN_g to represent the Bayesian network learned at the g th sample generation. The uniform random sampling is used to create ρ samples as the initial population, among which we select μ samples to construct the Bayesian network. The configuration samples with high performance evaluations get high chances to be selected. Once the Bayesian network BN_{g-1} is constructed, we generate λ new samples by sampling network and evaluate those samples for the system performance. The g th population of samples are then obtained by applying the recombination operator [1] on the λ new samples as well as the old ones. After that, we select μ samples from the new population again to construct Bayesian network BN_g for the next generation. Note the g th population of samples usually have better performance evaluations than previous populations because they are sampled from the latest Bayesian network that captures optimal regions in the configuration space. In addition, the update of Bayesian network will lead to a more accurate description of optimal regions in the configuration space, which in consequence produces samples with even better performances. Such iterative sample generation and network learning processes follow the strategies of evolutionary search, which will eventually converge to the optimal point in the configuration space[3].

Given: the sample population size ρ
two other parameters μ and λ with $u \leq \rho, \lambda \leq \rho$.

Generate the initial population with size ρ by uniform sampling.

Evaluate the samples.

Select μ samples from the population based on their performance evaluations to create BN_0 .

For generation $g = 1, 2, \dots$

Use BN_{g-1} to generate λ new samples;

Evaluate the samples;

Update the g th population with the new samples;

Select μ samples from the g th population based on their performance evaluations to create BN_g ;

End.

Figure 1: The configuration tuning based on Bayesian network construction and sampling.

Through the Bayesian network guided configuration tuning, we can discover the best configuration setting in S_0 . More importantly, we also obtain a Bayesian network as the byproduct of the tuning process. Such network can be utilized to benefit the configuration tuning in system S_1 . We still employ the algorithm in Figure 1 to conduct the optimal configuration search in system S_1 . However, rather than starting with the uniform random sampling to generate the initial population, we use the learned Bayesian network from S_0 to generate initial samples and build the initial network BN_0 for system S_1 . As a result, the generated initial samples in S_1 will be located more closely to the optimal area in the new configuration space. Furthermore, since the transferred Bayesian network records the dependencies of configuration parameters, a lot of unnecessary efforts for exploring the system configuration space can be avoided. The convergence speed of configuration search in S_1 can be significantly accelerated.

In the experiments, we first build a web based test bed system and denote it as S_0 . The Bayesian network based tuning algorithm is applied to discover the optimal configuration in S_0 as well as learn a Bayesian network that reveals the parameter dependencies

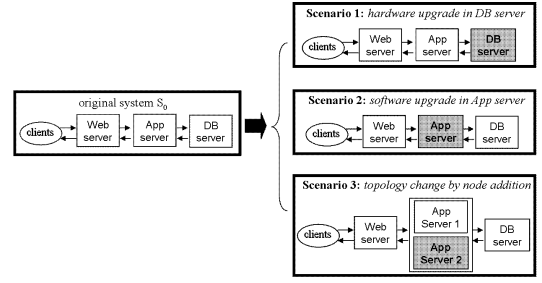


Figure 2: The original test bed system S_0 is upgraded by three different scenarios.

in S_0 . After that, we change the test bed system in three different ways, as shown in Figure 2, to reflect commonly encountered scenarios in real system evolutions: the hardware upgrade, the software version change, and the topology change by adding a node. In each upgraded system, we embed the Bayesian network learned in S_0 into the configuration tuning process, and compare the efficiencies of configuration searches with and without the help of transferred experiences. Table 1 summarizes the number of sample evaluations required to discover the optimal configuration as well as the best utility values produced by both search processes in all systems. It shows that the direct configuration search always takes more than 100 sample evaluations to reach its optimum whereas the experience guided search only needs tens of samples. In addition, the best utility discovered by the experience guided search is always higher than that obtained by the direct search.

	without transfer		with transfer	
	evaluations	utility	evaluations	utility
original system	132	9.11	-	-
hardware upgrade	109	12.25	25	12.27
software update	117	9.52	11	9.58
topology change	110	16.29	46	16.52

Table 1: Summary of experimental results.

3. CONCLUSIONS

This paper has proposed the experience transfer to improve the efficiencies of configuration tuning in computing systems. We have treated the dependencies between system configurations as useful experiences in configuration tuning, and proposed a Bayesian network guided tuning algorithm to discover the optimal configuration setting. Results have shown that our transferred dependency knowledge can achieve tremendous time savings for system operators in tuning the system performance.

4. REFERENCES

- [1] H. G. Beyer. *The theory of evolution strategies*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.
- [2] R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, April 2003.
- [3] M. Pelikan, D. E. Goldberg, and E. C. Paz. BOA: The Bayesian optimization algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '99)*, volume I, pages 525–532, Orlando, FL, 1999.
- [4] A. Saboori, G. Jiang, and H. Chen. Autotuning configurations in distributed systems for performance improvements using evolutionary strategies. In *28th IEEE International Conference on Distributed Computing Systems (ICDCS '08)*, 2008.
- [5] B. Xi, Z. Liu, M. Raghavachari, C.H. Xia, and L. Zhang. A smart hill-climbing algorithm for application server configuration. In *Proceedings of the 13th international conference on World Wide Web (WWW '04)*, pages 287–296, 2004.