Combining High Level Symptom Descriptions and Low Level State Information for Configuration Fault Diagnosis

Ni Lao – Tsinghua University;
Ji-Rong Wen and Wei-Ying Ma – Microsoft Research Asia
Yi-Min Wang – Microsoft Research

ABSTRACT

Automatic fault diagnosis is an important problem for system management. In this paper, we combine high level symptom descriptions and low level state information to solve the system fault diagnosis problem. We extract state-symptom correlation information from knowledge sources in text format, and then use symptom similarity to rank the candidate system states. We apply the method to Windows Registry problems to help Product Support Service (PSS) engineers. Promising results with two different knowledge sources show the robustness of our method. Finally, we explain why this combination is successful and also discuss its limitations.

Introduction

Configuration management will remain a persistent problem "as long as people change how they want to use the system" [Ande95]. Change and Configuration Management and Support (CCMS) of computer systems with large install bases and large numbers of available third-party software packages have proved to be daunting tasks [LC01]. Jim Gray depicted Trouble-Free System as an important goal of IT research: build a system used by millions of people each day, and yet administered and managed by a single part-time person [Gray03]. To achieve this goal, systems should be self-managing. Redstone and coworkers [RSB03] described a global-scale automated problem diagnosis system that collects problem symptoms from users' desktops, and then automatically searches global databases of problem symptoms and fixes. We address similar problems in a new way in this paper.

People typically use two different strategies to diagnose system faults: symptom-based approach and state-based approach. Nowadays, many systems have knowledge databases of their known problems online (such as [Apple], [BugNet], [MSKB] and [Redhat]). Computer users typically use symptom-based analysis to troubleshoot configuration problems. They describe their problems with words, and use information retrieval tools to find documents containing solutions to the problem. Considering that most customers are not PC experts, their problem descriptions are usually inaccurate and thus using them directly to retrieve relevant documents often yields unsatisfactory results.

At the other extreme, many tools attempt to automate the fault diagnosis task using low level machine states (such as [CKF02] and [Qie03]). Such tools usually provide a language to specify the expected behavior of the system, use monitors to detect system deviation from the rule, and define actions to correct them. For example, Strider [W03] uses various techniques to narrow down the list of candidate root causes, including persistent state differencing, runtime tracing, intersection and statistical ranking. It then uses configuration roll-back [SR00] to fix the problem. Unfortunately, in many cases, the ranking results are not satisfactory. Furthermore, its differencing step and tracing step are not always feasible.

In this study, we combine both high level symptom descriptions and low level state information to solve the configuration fault diagnosis problem. The idea is to extract correlation information between low level states and high level symptom from knowledge sources, and then use symptom similarity to rank the states. We apply the method to Windows Registry problems [Gan04]. Promising results with two different knowledge sources show the robustness of our method. Finally we try to explain why the combination is successful, and discuss its limitations.

System Architecture and User Scenario in PSS

The state-symptom correlation information required for our problem solving technique is extracted from various text-based knowledge sources, such as the Product Support Service (PSS) log and the Microsoft Knowledge Base (KB). The information is then stored in a database called the PC-Genomics Database. Figure 1 illustrates a scenario of how to use PC-Genomics technique for more effective problem troubleshooting.

For example a user named Diana cannot find any fonts in the font dialog box. This is because the registry keys that list TrueType fonts are damaged. The processes from step (1) through step (7) show how her problem is solved:

- **Step 1**: Diana reports the problem to PSS. She goes to http://support.microsoft.com and describes the problem with a short paragraph.
- **Step 2**: The PSS engineer initially tries to diagnose the problem using the normal method. If this works, go directly to step 7.
- **Step 3**: The state collection and analysis tools are downloaded from the site to Diana's machine.
- **Step 4**: Diana runs Strider to compare bad states and good states in Restore points, and the trace log is also produced.
- **Step 5**: A candidate set containing possible incorrect states is generated from this collected data and sent back to PSS.
- **Step 6**: The candidate set is fed in to PC Genomics database to figure out the root cause of the problem.
- **Step 7**: The generated solution is sent back to the Diana. It could be either a solution script, an executable or related KB articles. In this case, she receives a solution script which deletes the key: key_local_machine\software\microsoft\windows nt\currentversion\fonts.

Extracting State-Symptom Correlation from Knowledge Sources

The PC-Genomics Database

Obtaining the specific data needed for the PC-Genomics Database requires different techniques for different data sources. The digital knowledge sources we use usually consist of articles in free text form. Although we are far from being able to understand the meaning of these articles automatically, we can identify state names and the portion of text which is specifying the problem symptom in these articles. This state-symptom co-occurrence is crucial information to link states with their symptoms. Sometimes the corresponding software name and resolution can also be identified, and the data extracted for the PC-Genomics Database will have the form of Table 1.

The Registry Dictionary

The Windows Registry is the main configuration state store on PCs. It has a tree like structure, and each piece of configuration state is specified by a path name and optionally a value name. Either a path name or a value name prefixed by its path name is called an entry, and there are typically more than 200,000 registry entries on a machine [SR]. To locate registry entries within free format text, we first collected all the registry entries from 50 PCs. They contained 898,546 unique registry entries after name canonicalization (e.g., substituting different user IDs, like "s-1-5-21-. . ." with the string "UID" in the registry entry path). Then they are used as a "registry dictionary" to help recognize registry entries within free format text.

The Knowledge Sources

The PSS log is an archive of problem-solving cases maintained by Microsoft Corporation. Each case contains the exchanged emails between a customer and a support engineer (see Figure 2). The total PSS

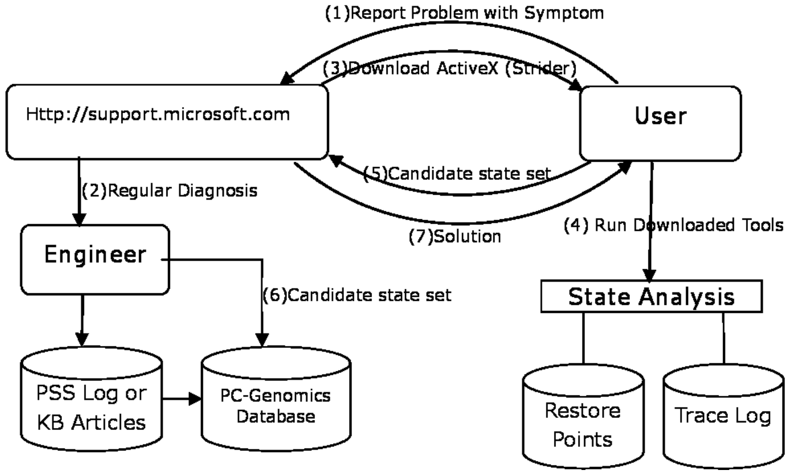
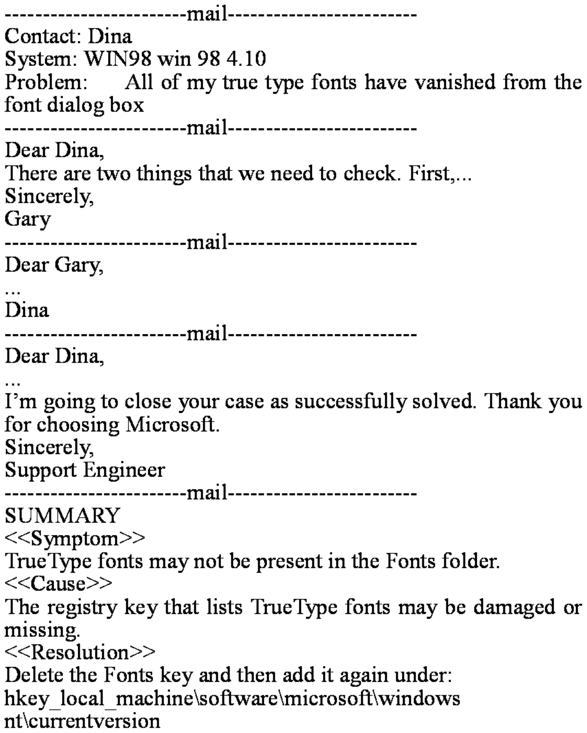


Figure 1: Architecture of PC-Genomics troubleshooting.

ID	State (Registry Key)	Symptom	Software	Solution
1	HKLM\Software\Policies\Microsoft\Messenger\Client	Can not run Windows Messenger. . .	Messenger	delete PreventRunregister item
2	HKLM\Software\classes\clsid\{f414-a00bb8}\inprocserver32	System Restore GUI is blank. . .	Windows	1. Go to "Start→Run" 2. . . .

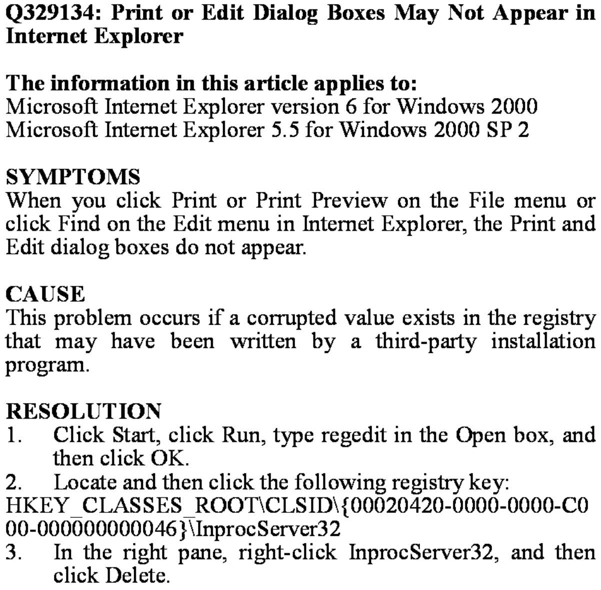
Table 1: Format of the PC genomics database.

log body contains more than 10 million cases. We used 2,311,492 cases in our experiment. These cases cover 15 products within six product families, ranging in time from 3/20/1997 to 5/13/2003 (see Table 2). We combine the *action* and *result* part of the summary as the symptom of a case. If a registry key is referenced in the final mail message of a case, this entry and the case symptom were added into the PC-Genomics Database as a pair. We found that 143,157 of PSS log cases referenced 4,837 unique registry entries, and 1,913 of them are registry values.



```
------------------------mail------------------------
Contact: Dina
System: WIN98 win 98 4.10
Problem:     All of my true type fonts have vanished from the
font dialog box
------------------------mail------------------------
Dear Dina,
There are two things that we need to check. First,...
Sincerely,
Gary
------------------------mail------------------------
Dear Gary,
...
Dina
------------------------mail------------------------
Dear Dina,
...
I'm going to close your case as successfully solved. Thank you
for choosing Microsoft.
Sincerely,
Support Engineer
------------------------mail------------------------
SUMMARY
<<Symptom>>
TrueType fonts may not be present in the Fonts folder.
<<Cause>>
The registry key that lists TrueType fonts may be damaged or
missing.
<<Resolution>>
Delete the Fonts key and then add it again under:
hkey_local_machine\software\microsoft\windows
nt\currentversion
```

Figure 2: Sample emails in PSS log.



Q329134: Print or Edit Dialog Boxes May Not Appear in Internet Explorer

The information in this article applies to:
Microsoft Internet Explorer version 6 for Windows 2000
Microsoft Internet Explorer 5.5 for Windows 2000 SP 2

SYMPTOMS
When you click Print or Print Preview on the File menu or click Find on the Edit menu in Internet Explorer, the Print and Edit dialog boxes do not appear.

CAUSE
This problem occurs if a corrupted value exists in the registry that may have been written by a third-party installation program.

RESOLUTION
1. Click Start, click Run, type regedit in the Open box, and then click OK.
2. Locate and then click the following registry key:
HKEY_CLASSES_ROOT\CLSID\{00020420-0000-0000-C000-000000000046}\InprocServer32
3. In the right pane, right-click InprocServer32, and then click Delete.

Figure 3: A Sample KB Article.

The Microsoft Knowledge Base [MSKB] contains troubleshooting articles written by experienced engineers (see Figure 3). Our data consist of 142,448 articles ranging from Q10022 to Q332210. The KB articles are written in well formed XML format, so it is easy to parse their symptom section. A registry key found anywhere in the article is added into the PC-Genomics Database with the corresponding symptom. We found that 1,921 of the KB articles reference 996 unique registry entries and 412 of them are registry values.

Rank States Using State-Symptom Correlations

Symptom Similarity

A state-based tool, like Strider, can generate a set of states as candidates for the root cause of a given problem. Our approach matches the symptom of the

Product Name	Product Family Name	Case Count
Office 2000 Win32 EN	Office	88,136
Office SBE 2000 Win32 EN	Office	25,507
Office Pro 2000 Win32 EN	Office	245,429
Office Prem 2000 Win32 EN	Office	149,722
Office Pro XP Win32 EN	Office	80,615
Visual Basic Enter Win 5.0 EN	Visual Basic	17,746
VB Enterprise 6.0 Win32 EN	Visual Basic	38,917
SQL Srvr 7.0 WinNT EN BETA	SQL Server	30,445
SQL Server Ent 7.0 WinNT EN	SQL Server	33,040
Windows Advanced Svr 2000 EN	Windows NT	48,174
Windows Svr 2000 EN	Windows NT	155,892
Exchange Server 5.5 EN	Exchange	251,974
Exchange Svr 2000 EN	Exchange	99,608
Windows XP Home Edition EN	Windows XP	707,908
Windows XP Professional EN	Windows XP	338,379

Table 2: The PSS log used as a knowledge source.

problem with those symptoms of each candidate state in the PC-Genomic database. In this way, we can estimate the similarity between current problem and recorded previous problems (see Figure 4). We employ traditional *tf-idf* and *Cosine* [V79] measures from information retrieval to calculate similarity values. In the database, all the symptoms of a root cause are combined as a mixed symptom $S_i = S_{i,1} + S_{i,2} + \cdots + S_{i,n}$. The current symptom $S_{current}$ or each of the recorded symptoms S_i is represented as a vector of term frequency. The basic assumption here is that if a state is a good candidate to the current problem, it is highly likely that it caused some problems with similar symptoms in the past. The calculated similarity values are used to rank the candidate registry entries.

Intersection-Ranking, Diff-Ranking & Trace-Ranking

We collected 74 registry-related real-world problems reported by our colleagues and users of web support forums. These problems are independent from the PC-Genomics Database we are building. For each problem, we recorded its symptom description, trace set, and differencing set. We also calculated the intersection set and root cause rankings with the Strider tool. There are three points to apply our ranking schemes. First, we can apply ranking on the intersection-set and it is called *intersection-ranking*. This ranking is expected to produce better result since the size of intersection is relatively small and the ranking cost is also small. Second, when no trace data is available, we can directly apply *diff-ranking* to the diffing set. Finally, when no diffing data is available, we can also directly rank the trace set, which is called *trace-ranking*.

Relaxed Root Cause Matching in the Database

If both the value name and path name of a root cause can be matched in the PC-Genomics database, we call it *value-matched*. If only the path name is matched, we call it path-matched. Otherwise, we call it not-matched. For example, the configuration state with path name "hkey_classes_root\.jpg," and value name "(Default)" is considered value-matched, if "hkey_classes_root\.jpg\(Default)" can be found in the database. Or else if "hkey_classes_root\.jpg" is in the database, it is called path-matched. If none of them are in the data base, this configure is considered to be not-matched. To our experience, this relaxed matching criterion can increase the problem coverage of our method and do little harm to its accuracy.

Because the "registry dictionary" covers only 66 of the 74 root causes, the remaining eight root causes could not be recognized in the knowledge source free text. The PC Genomic database extracted from PSS log covers the root causes of 59 problems, while the database from KB covers 37.

Ranking Result

The result of a diagnosis processes is actually a rank of candidate root causes, ordered by their likelihood of being the actual one. Obviously, the real root cause should be ranked as high as possible in order for this approach to be effective. Usually, a ranking less than 5 is preferred. For each method, we sorted the cases by the ranking of their actual root causes. With these ranking curves, we can easily compare the diagnostic effectiveness of different methods.

Figure 5 contains the ranking curves using PSS log. We can see that our method efficiently increases the diagnosis accuracy with intersection data. Only nine out of 59 real root-causes rank more than 5. However, ranking with only differencing data or trace data is not very accurate.

Figure 6 contains the ranking curves using KB. The ranking curves show that our knowledge database from KB articles still gives good accuracy with intersection data and differencing data, but the ranking with trace data deteriorates a lot.

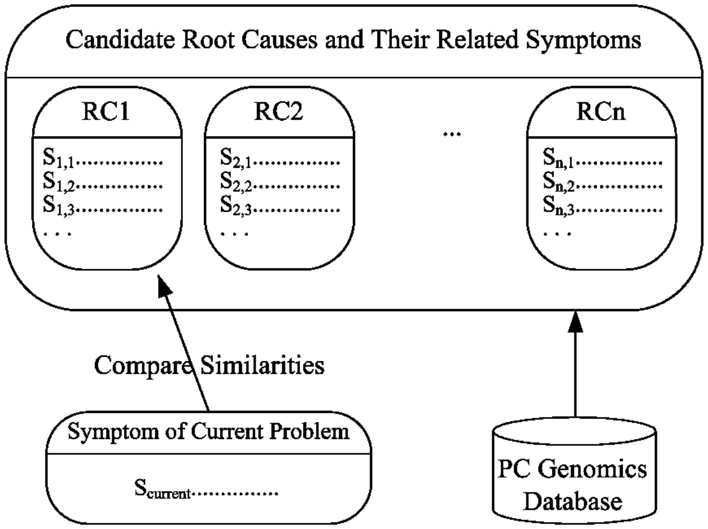


Figure 4: Symptom similarity for root cause ranking.

Discussion

In this section, we discuss the strengths and weaknesses of the method.

One-to-many Mappings Between Symptoms and States

Problems with the same or similar symptom(s) may be caused by different registry entries. For instance, we manually checked the PSS log and found that 17 entries have been reported to cause the "Cannot open Word document" problem (see Table 3). If we use only the symptom-based methods for troubleshooting, we will get multiple possible root causes for a problem. In our approach, state information, like a filter that is orthogonal to the symptom description, can point out the root cause efficiently.

Problem Coverage

The effectiveness of our approach depends on the problem coverage of our database. We need two things to achieve this goal: building a "registry dictionary" with good problem coverage, and extracting information from a knowledge source of good problem coverage.

In PSS log data, only about 0.5% registry entries are ever reported to cause problems (i.e., 4,837 of 898,546). In KB data, only about 0.1% entries are ever reported (i.e., 996 of 898,546). If we consider these entries as a filter, they would be very efficient in scaling down the candidate root cause set.

Among the 4,837 fragile registry entries from the PSS log, only a few entries cause problems frequently. Most entries have small numbers of occurrences (see Figure 7). They approximately follow a Zipf distribution [Z49]. Even if the PC-Genomics Database contains only a portion of the known problems, it can still greatly reduce the real-world enterprise support cost because the most costly problems are well covered.

Since the state-symptom database is flexible, the coverage problem can be further alleviated. If we find

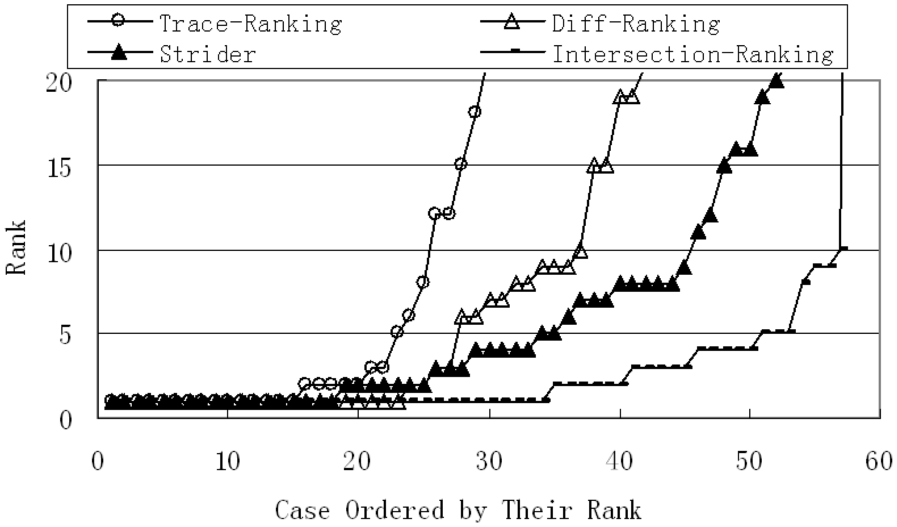


Figure 5: PSS log ranking results.

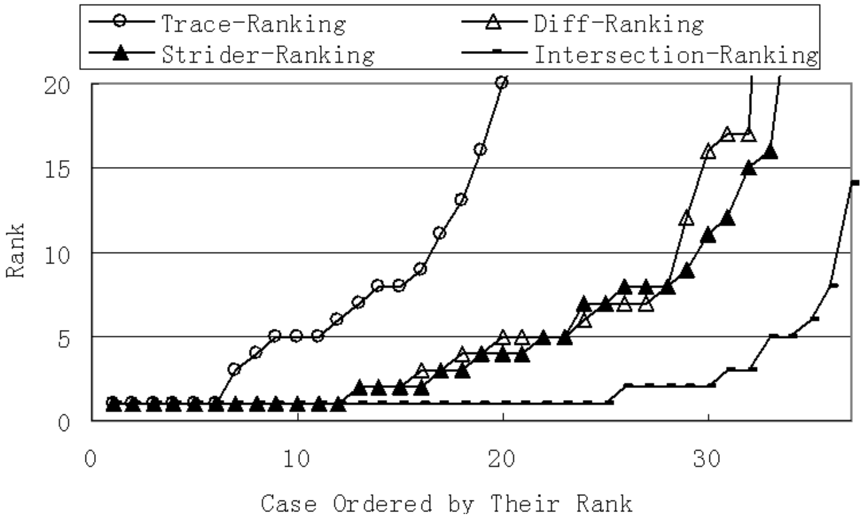


Figure 6: KB ranking results.

a common problem outside the dictionary or knowledge source, we can simply add its symptom and state into the database manually. But as long as we can only handle problems for which the root-cause entries have been found before, Strider needs to be used to find those root causes for the first time.

The Percentage of Registry-Related Problems

Directly calculating the percentage of registry-related problems from the number of cases which have cited a registry key yields a very small number: 6.2% (i.e., 143,157 of 2,311,492). One reason behind this is that the engineers often cite a registry related KB article and ask the user to do what the article says without specifying the registry key name. Another big reason is that finding root causes of Registry problems were extremely hard before Strider.

With the help of KB information for each case, we can get a better estimation. When a case is closed, the engineer is required to cite a KB article ID as the description of the case. About 8.8% of KB articles (i.e., 12,464 of 142,448) contain the keyword "registry." These "registry" KB articles cover 27% of the cases that ever cite KB. We manually verified 40 of these KB articles, and found that 15 of them are not actually registry problems. They may be just providing registry-related knowledge or using registry as a problem solving method. Thus, the overall registry problem percentage is approximately 17% (i.e., 27%*25/40).

Summary

We have proposed a novel solution to combine the traditional symptom-based troubleshooting method and relatively new state-based troubleshooting method. It adds some overhead of data collection to the user, but it can solve some previously hard-to-solve problems. So we prefer to treat the PC-

hkey_users\.default\Software\Microsoft\Office\8.0\Outlook\Options\Mail
hkey_current_user\software\microsoft\office\9.0\word\data\toolbars
hkey_current_user\software\microsoft\office\9.0\word\data\settings
hkey_current_user\software\microsoft\office\10.0\word\data\settings
hkey_current_user\software\microsoft\office\10.0\word\data\toolbars
hkey_local_machine\SOFTWARE\Microsoft\Internet Explorer\Plugin
hkey_current_user\environment
hkey_local_machine\System\CurrentControlSet\Services\Inetinfo\Parameters\MIMEMap
hkey_classes_root\.doc\Content Type
hkey_classes_root\mime\database\content type
hkey_classes_root\MIME\DATABASE\Charset
hkey_classes_root\MIME\DATABASE\Codepage
hkey_classes_root\word.document
hkey_classes_root\excel.sheet.8\shell\open\command
hkey_current_user\software\microsoft\office\9.0\common\general\startup
hkey_local_machine\software\microsoft\shared tools\text converters\import
hkey_local_machine\software\microsoft\shared tools\text converters\import\wordperfect6\&x
hkey_classes_root\excel.sheet.8\shell\open\ddeexec
hkey_current_user\software\microsoft\shared tools\outlook\journaling\microsoft word\autojournaled

Table 3: Registry key root causes of "Cannot open Word document" problem.

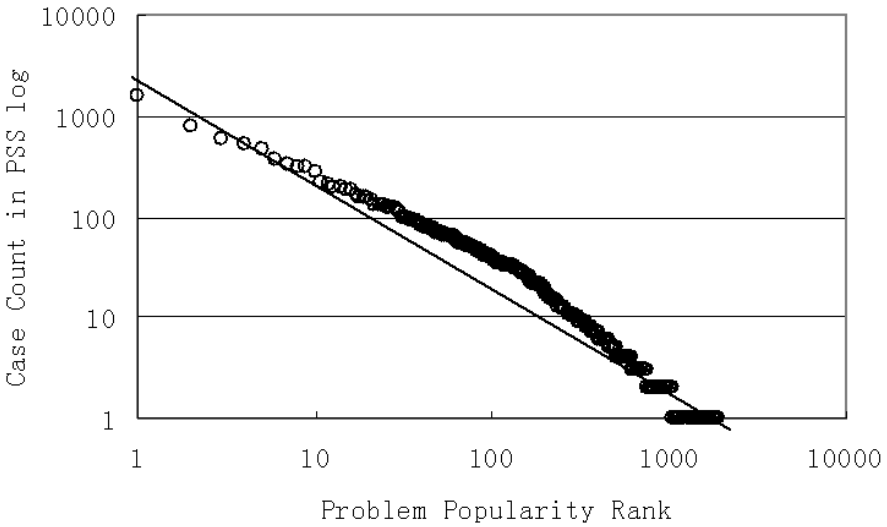


Figure 7: Occurrence of 1,913 value-matched registry entries in the PSS Log.

Genomics technique as a backup for the regular methods and to use it only when regular methods fail. Our future work will exploit other types of system information which can give the troubleshooting process better problem coverage and make it more automatic.

Acknowledgement

would like to express our sincere thanks to our shepherd AEleen Frisch for her valuable feedback, to Aaron Johnson, Chad Verbowski, and Archana Ganapathi for their analysis of the test cases, and to the many colleagues who helped collect the registry snapshots from 50 computers.

Author Information

Ni Lao is currently a graduate student in School of Software at Tsinghua University in China. He received his B.S. in Electronic Engineering from Tsinghua University in 2003. His current research focuses on automated system management using data mining and pattern recognition methods. He can be reached at noon99@mails.tsinghua.edu.cn.

Ji-Rong Wen is a researcher in Microsoft Research Asia. He received his Ph.D. in Computer Science from the Institute of Computing Technology, the Chinese Academy of Science in 1999. He joined Microsoft in July 1999. His current research interests are data management, information retrieval (especially Web search), data mining and system management.

Wei-Ying Ma received the B.S. degree in electrical engineering from the national Tsing Hua University in Taiwan in 1990, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of California at Santa Barbara in 1994 and 1997, respectively. He joined Microsoft Research Asia in April 2001 as the Research Manager of the Information Management and Systems Group. Prior to joining Microsoft, he was with Hewlett-Packard Laboratories at Palo Alto. From 1994 to 1997 he was engaged in the Alexandria Digital Library (ADL) project in University of California at Santa Barbara while completing his Ph.D. Dr. Wei-Ying Ma serves as an Editor for the ACM Multimedia System Journal and Associate Editor for the Journal of Multimedia Tools and Applications published by Kluwer Academic Publishers. His research interests include Internet search, information retrieval, content-based image retrieval, intelligent information systems, adaptive content delivery, and media distribution and services networks.

Yi-Min Wang is the manager of the Systems Management Research Group at Microsoft Research, Redmond. He received his Ph.D. in Electrical and Computer Engineering from University of Illinois at Urbana-Champaign in 1993, worked at AT&T Bell Labs from 1993 to 1997, and joined Microsoft in 1998. His research interests include systems and security management, fault tolerance, home networking, and distributed systems.

References

[Ande95] Anderson, E. and D. Patterson, "A Retrospective on Twelve Years of LISA Proceedings," *Proceedings of the Thirteenth Systems Administration Conference (LISA XIII),* USENIX, p. 95, 1999.

[Apple] *Knowledge Base*, http://kbase.info.apple.com .

[BugNet] BugNet, *BugNet*, http://www.bugnet.com .

[CKF02] Chen, M., E. Kiciman, E. Fratkin, A. Fox, and E. Brewer, "Pinpoint: Problem Determination in Large, Dynamic, Internet Services," *Proc. Int. Conf. on Dependable Systems and Networks (IPDS Track)*, 2002.

[Gan04] Ganapathi, A., Yi-Min Wang, Ni Lao, and Ji-Rong Wen, "Why PCs Are Fragile and What We Can Do About It: A Study of Windows Registry Problems," to appear in *Proc. IEEE DSN/DCC*, June 2004.

[Gray03] Gray, J., "What Next? A Dozen Information-Technology Research Goals," *Journal of the ACM*, Vol. 50, Num. 1, pp. 41-57, January 2003.

[LC01] Larsson, M. and I. Crnkovic, "Configuration Management for Component-based Systems," *Proc. Int. Conf. on Software Engineering (ICSE)*, May 2001.

[MSKB] Microsoft Corporation, *Microsoft Knowledge Base*, http://support.microsoft.com .

[Qie03] Qie, X.-H., Sanjai Narain, "Using Service Grammar to Diagnose BGP Configuration Errors," *Proc. Usenix Large Installation Systems Administration (LISA) Conference*, pp. 237-246, October 2003.

[Redhat] Redhat Corporation, *Redhat Support Forums*, http://www.redhat.com/support/forums .

[RSB03] Redstone, J. A., M. M. Swift, B. N. Bershad, "Using Computers to Diagnose Computer Problems," *Proc. HotOS*, 2003.

[SR] Windows XP System Restore, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwxp/html/windowsxpsystemrestore.asp .

[SR00] Solomon, D. A. and M. Russinovich, *Inside Microsoft Windows 2000*, Microsoft Press, Third Edition, Sept 2000.

[V79] van Rijsbergen, C. J., *Information retrieval*, Butterworths, Second Edition, London, 1979.

[W03] Wang, Y.-M., Verbowski, Chad., Dunagan, J., Chen, Y., Wang, H. J., Yuan, C., and Zhang, Z., "STRIDER: A Black-box, State-based Approach to Change and Configuration Management and Support," *Proc. Usenix Large Installation Systems Administration (LISA) Conference*, pp. 159-171, October 2003.

[Z49] Zipf, G. K., *Human Behavior and Principle of Least Effort: An Introduction to Human Ecology*," Addison Wesley, Cambridge, MA, 1949.