Making Sense of Thousands of Trace Files

If you do it right, the overhead incurred by Oracle tracing is imperceptible, so more sites than ever are using trace data as a source for detailed performance data. But what can you do when you end up with thousands of trace files? With Method R Workbench, it's not a problem.

by Cary Millsap

Oracle tracing is an economical but rich performance data source.

If you do it right, the overhead incurred by tracing is imperceptible.

These factors conspire to make tracing more attractive than ever, which can result in enormous volumes of data spread across thousands of files.

Method R Workbench can process thousands of trace files in just a few minutes.

This opens new possibilities like digging trace data for individual user experiences out of hard-to-trace applications, measuring the health of your system as your users perceive it, and understanding complex workload flow with Gantt charts.

Problem

How do I fix a slow program? No matter whether it's early in your software development life cycle or late in production, the answer is the same. I begin by tracing it, so I can see how the program you care about has spent its time.

Some applications have tracing features built in that make targeted tracing easy. For example, if a program sets its Oracle user session handle attributes to identifying values, then you can trace that program with laser beam precision, resulting in a single trace file that contains exactly the information you want.

But when an application doesn't identify its programs this way, then it's harder to trace just the program you're interested in. An easy solution is to trace a whole service, or maybe even a whole instance. It's easy to trace everything, but then you're just kicking the identification problem down the road: how are you going to sift through a morass of data to isolate just the trace data you need?

In case you're wondering, we used to worry that tracing lots of concurrent sessions would put too much overhead on a system. But experience has shown that if you trace correctly (for starters, using `wait=true`, `bind=false`, and `plan_stat=first_execution`), you can trace even a whole database for hours at a time, and—as long as you don't run out of space—nobody even notices.

So "big tracing" is a legitimate data collection tactic. But then you end up with thousands of trace files. How can you make sense of all the data?

Plan

Making sense out of thousands of *anything* takes software. For decades, nobody made software to help manage lots of trace files. But today, our Method R Workbench application gives you the features you need for doing "big trace" work. With Method R Workbench, you can:

» See general information about your trace files, like the durations of the experiences they represent, and the start and end times of those experiences.

» See the relationships in time among the trace files you have.

» Sift through thousands of trace files to find the one that contains the user experience you're interested in analyzing.

» Crop out just the trace data for that targeted user experience, from within a file

that contains additional data that you don't care about.

Analysis

A core feature of Method R Workbench is our trace file cropper. It automates three important tasks that form the foundation for working with lots of big trace files:

» Crop just one interesting segment of trace data out of a big trace file.

» Explode a trace file representing lots of user experiences into lots of smaller files, each one representing an individual user experience.

» Discard trace files that don't participate in a specified time interval.

Here's an example. Imagine that you have a program that ran way longer than normal between 13:07:10 and 13:09:28 yesterday. You have thousands of trace files collected around that observation interval, but you don't know which file contains the trace for your execution. And, because your application uses connection pooling, it's likely that the trace file you want will contain data from other programs that you're not interested in analyzing today, so you don't want a *whole* trace file, just the relevant part of one.

A few years ago, it would have sounded impossible, but Method R Workbench makes it easy to find your interesting program execution from a morass of data with just one command. That one command (it's `mrcrop island`, if you're curious) can explode your trace files into individual user experiences and discard the ones that don't participate in your 13:07:10–13:09:28 time interval. In the few files that remain, you'll be able to identify your trace file by its start and end times, or by recognizing its SQL statements. The first time you do it, it feels like a miracle.

Solution

Projects that used to really stretch us are no trouble now.

» Not long ago, a client wanting to diagnose a slow program traced a whole instance and sent us 29,505 trace files. It took us several hours of meticulous labor to figure out that only *three* of these files contained data that we cared about. With today's Method R Workbench, we would find their slow program's trace data within just a few minutes of receiving the files.

» So-called health checks have bothered me since the mid-1990s when I realized that a system with an A+ health check score isn't necessarily a healthy system. But with trace data as your primary data source, you can view your system as your users perceive it, in finer resolution than with any other data source. You'll be able to see not just the big problems people are feeling, but also the creeping inefficiencies that nobody has even noticed yet.

» Optimizing complex batch workloads can be tricky. AWR doesn't help you see your code's execution flow through time, and the core problem is often not SQL or PL/SQL code that could show up in your top ten lists. But it's easy to load an entire batch window's trace files into Method R Workbench and display the data as a Gantt chart in Microsoft Excel. The Gantt chart is the perfect tool for seeing your workload's critical path, so you can target your analysis on the right trace files. And with Method R Workbench, you can drill all the way down to a single database call or system call whenever you need to.

Technology

Method R Workbench is easy-to-use, high-precision *Oracle time measurement software* for software development, code reviews, performance tests, concept proofs, hardware and software evaluations, upgrades, troubleshooting, and more—for Oracle developers, DBAs, and decision-makers in every phase of the software life cycle. The cropping tool and Gantt chart export function described in this monograph are standard features that are shipped with the product.

Method R Workbench 9

◇ Method R™
method-r.com
info@method-r.com