AMPRA: Analyzer of Memory Protection and Failures Implications on TCO

Documentation

Panagiota Nikolaou, Yiannakis Sazeides, Lorena Ndreu, Marios Kleanthous Ξ - Computer Architecture Research Group University of Cyprus

The research leading to this tool is supported by the European FP7 project "Harnessing Performance Variability" (FP7-ICT-2013-10-612069, HARPA).

Email Contact: panagiota.nikolaou@cs.ucy.ac.cy

Tool site: http://www2.cs.ucy.ac.cy/carch/xi/ampra_tco.php

Project: http://www.harpa-project.eu/

European Commission FP7: http://cordis.europa.eu/fp7/home_en.html

Revision Control

Ver 1.0	2/12/15	Panagiota Nikolaou

AMPRA

This tool models the implications of DRAM errors and DRAM protection techniques on the total cost of ownership of a data center populated by different resources (computing nodes, storage nodes…).

Basic Operation

The AMPRA tool takes as input the raw failure rates, the configuration of each protection technique including power and temperature for different scenarios and a data center configuration and one resource configuration and the protection technique that you want to investigate and estimates the TCO breakdown of that configuration.

The tool is defined to explore trade-offs between different protection techniques.

Download & Install

Requirements:

- Perl, g++ compiler, Bash, linux environment, libstdc++.so.6

To install libstdc++.so.6 for Fedora use yum install libstdc++-devel.i686

Download the tool from http://www2.cs.ucy.ac.cy/carch/xi/ampra_tco.php

The tool is written in Perl, Bash and in C++

The AMPRA tool is distributed in three main parts:

1. The Calculation of the propagation of DRAM errors from given raw FITs
2. The Calculation of the TCO with a datacenter configuration
3. The wrapper that is used to encapsulate all the parameters and investigate the TCO for a given protection and different workloads

Use

Before running the tool you have to execute the following:

chmod +x makeAMPRA

./makeAMPRA

Run the tool by writing:

Please note that you have to be in the AMPRA directory to run the tool, otherwise you will get some errors.

./super_script.pl {protection_technique} {scenario}

protection_technique is a DRAM protection technique that you want to investigate.

The protection technique can be ChipkillDC, ChipkillSC and SECDED.

scenario is the online, or offline or collocated scenario that you want to run

Scenario selection determine the input power, temperature and performance that are estimated for a given protection technique in real hardware using Intel Xeon E5620 servers located to the University of Cyprus. You can find the input server power, server performance and component temperature for each technique that is determined running in real hardware in this directory input_example/ power_temp_perf_file_{protection}.

Scenarios can be: 2Web, 2Web1Mcf, 2Web2Mcf, 2Web1Gcc, 2Web2Gcc, 2Web1Stream, 2Web2Stream, 2Web1Hadoop500, 2Web2Hadoop500, 2Web1Hadoop49000, 2Web2Hadoop49000

2Web: Results from running 2 Web Search in two cores

2Web1Mcf: Results from running 2 Web Search in two cores and 1 MCF in one core

2Web2Mcf: Results from running 2 Web Search in two cores and 2 MCF in two cores

2Web1Gcc: Results from running 2 Web Search in two cores and 1 Gcc in one core

2Web2Gcc: Results from running 2 Web Search in two cores and 2 Gcc in two cores

2Web1Stream: Results from running 2 Web Search in two cores and 1 Streaming benchmark in one core

2Web2Stream: Results from running 2 Web Search in two cores and 2 Streaming benchmark in two cores

2Web1Hadoop500: Results from running 2 Web Search in two cores and 1 MapReduce 500MB training size in one core

2Web2Hadoop500: Results from running 2 Web Search in two cores and 2 MapReduce 500MB training size in two core

2Web1Hadoop49000: Results from running 2 Web Search in two cores and 1 MapReduce 49000MB training size in one core

2Web2Hadoop49000: Results from running 2 Web Search in two cores and 2 MapReduce 49000MB training size in two core

The output of the tool shows extensive information about costs and other estimations concerning the data center.

Note that to extend the tool with more protection techniques you have to modify super_script.pl and add the power_temp input file.

Structure

The AMPRA tool is written in Perl, C++ and Bash organized in 14 files: + makeAMPRA + super_script.pl + AMPRA.pl + temp_model.pl + mttf.pl + cost-et.pl + ChipkillSCfailures/ChipkillSC_prob_wrap.pl + ChipkillSCfailures/ChipkillSC_prob + ChipkillDCfailures/ChipkillDC_prob_wrap.pl + ChipkillDCfailures/ChipkillDC_prob + SECDEDfailures/SECDED_prob_wrap.pl + SECDEDfailures/SECDED_prob + packages/resource.pm + packages/dc.pm +

The makeAMPRA is the first script that is used to change mode to all the scripts and to delete some unnecessary files.

The super_script.pl is the script that is used to run the tool with a specific protection technique and a scenario. First, runs a C++ program to estimate how the raw Fits are estimated for each protection technique. The C++ programs are located to the following directories according to the protection that is selected: 1. ChipkillSCfailures, 2. ChipkillDCfailures and 3. SECDEDfailures

ChipkillSCfailures folder contains a wrapper called ChipkillSC_prob_wrap.pl to a C++ program called ChipkillSC_prob. ChipkillSC_prob_wrap.pl wrapper takes an input file and transforms it to the different arguments that the ChipkillSC_prob program takes to run the analysis that estimates the propagation of different raw failures to DUE (detectable and uncorrectable), CE (correctable) and NDE (non detectable) errors. Accordingly, all the other folders contain the same programs but for the remaining protections (ChipkillDC and SECDED).

Then when the propagated fits are estimated the script calls the AMPRA tool to estimate the TCO for a given configuration with the given Fits.

The AMPRA tool is a wrapper to the main cost-et tool. This script takes as input the FIT rates, power, temperature and performance for a protection technique and a scenario and estimates the extra servers that are needed to keep the availability of a server and also the new MTTF. To

calculate the new MTTF for each component except of DRAM it calls the temp_model.pl and mttf.pl script.

The temp_model.pl script uses the Arrhenius function to calculate with a given reference and actual temperature the new MTTF of a component. Then it calls the mttf.pl to estimate the new mttf of the component considering the utilization parameter. For DRAM component first AMPRA script estimates the Fits for different repair techniques and then calls the mttf.pl script to estimate the new MTTF according to the temperature.

Then cost-et.pl script uses power and performance numbers and also the new MTTF for each component and the extra servers to estimate and output the TCO breakdown along with other intermediate results

dc.pm and resource.pm are packages used by cost-et.pl script to parse configuration files, to store the inputs information, to print the configuration, to estimate TCO and intermediate results. In the main directory, there are two subdirectories: + input_example which contains configuration file examples + packages which contains dc.pm and resource.pm.