

# A Spectrum of IV&V Modeling Techniques

## Implementation of Metrics-Collection Methods<sup>1</sup>

We have implemented Metrics-Collection methods described in the related deliverable document, “Definition of Metrics-Collection Methods,” and compared the performance of model-based tools with preliminary experiments ([lurch.pdf](#)—this paper describes Lurch and preliminary experiments in which metrics were collected to compare Lurch’s performance to existing tools).

### Preliminary Experiments

For these experiments execution time, peak memory use, and accuracy were measured for our random search verification tool, Lurch, and for the popular model checking tools SPIN and SMV. Execution time and memory were measured externally (we did not rely on resource use information supplied by the tools). Time was measured using the standard cygwin (UNIX emulator for Windows) ‘time()’ utility. For Lurch and SPIN, which require setup steps before the final verification step is run, all setup steps were timed and added to the total time for each run of the experiment. Memory was measured using the Windows XP “typeperf” utility, from which logs of peak memory use were collected for Lurch, SMV, and SPIN, throughout the experiment.

The preliminary experiment was based on simple tic-tac-toe games, which, while they represent large and difficult combinatorial problems for the verification tools, have the advantage that they can very easily be checked by hand (see [lurch.pdf](#)). For the accuracy comparison, results were checked by hand for false positives; that is, where all tools agreed no error was present in the model, the model was checked by hand to see if the error was present. Conventional model checking tools perform a search of *possible* behavior—they are not capable of finding behavior which is not possible for the model, so it is not necessary to check for false negatives when the verification terminates normally. Theoretically this should be true of Lurch also. In order to make sure we checked that Lurch always agreed with SMV and SPIN (the complete search tools) when they terminated normally.

---

<sup>1</sup> Because this document was delivered behind schedule, it contains information up-to-date 3/03, and there is significant overlap with the document “Definition of Metrics Collection Methods” delivered at the same time.

## Case Study Metrics

For the case study (flight guidance system) models provided by the University of Minnesota, we will compare Lurch's performance to SMV in much the same way Lurch was compared to SMV and SPIN in our preliminary experiments (see [case\\_study.ppt](#) for an overview of the case study). We will track execution time and peak memory use for Lurch, running on input models translated from RSML<sup>e</sup> to Lurch's input language, and for SMV, running on RSML<sup>e</sup> models by way of the University of Minnesota's Nimbus toolset.

In this case there is no simple solution oracle (as with the tic-tac-toe game models in the preliminary experiments). Lurch's accuracy will be measured in the following way: models with seeded faults will be provided by UMN, along with a list of properties, some of which will be violated by particular models, depending on the seeded faults. Lurch will be run on those models, and then the results of those runs will be compared to SMV results (already known to be true) by UMN.

Based on our preliminary experiments, we expect to see that Lurch will be as accurate (or perhaps not quite as accurate) as SMV, but will work faster and with much less memory.