WHAT IS CLAIMED IS:

1. A method of automatically tuning a software application, comprising:

a) modifying execution of the software application using a first parameter;

b) scoring the first parameter based on a log value from the software application and an improvement goal;

c) storing the first parameter, the score of the first parameter and the log value in a data store;

d) combining the first parameter with a selected parameter retrieved from the data store, the combining resulting in a second parameter;

e) repeating (a) through (d) until a criteria is met; and

f) when the criteria is met, generating tuning results based on the parameters, the log values and the improvement goal.

2. The method of claim 1, further comprising before (b), validating the log value.

3. The method of claim 1, wherein (b) comprises scoring the first parameter based on a log value from the software application, a log value from the data store, and the improvement goal.

4. The method of claim 1, wherein the combining is performed using a genetic algorithm.

5. The method of claim 1, wherein the criteria is met when all appropriate parameter values have been used to modify the execution of the software application.

6. The method of claim 1, wherein the criteria is met when, over a period of time, the incremental improvement resulting from applied parameter sets does not exceed a predetermined threshold.

7. The method of claim 1, wherein a parameter can include a combination of application settings.

8. The method of claim 1 wherein the software application is a virtual machine.

9. The method of claim 1, wherein the log value is one of pause time, throughput or memory footprint.

10. The method of claim 1, wherein the improvement goal is reducing application latency.

11. The method of claim 1, wherein the selected parameter retrieved from the data store in (d) is selected based on a score of the parameter.

12. A system to automatically tune a software application, comprising:

an extractor configured to receive log values from an operation of the software application, the operation of the software application being affected by a first parameter set;

a data store configured to store the received log values and the first parameter set;

a fitness determiner, configured to score the first parameter set based on the received log values and a goal;

a hypothesizer configured to retrieve a selected parameter set from the data store and combine the first parameter set with the selected parameter set to produce a second parameter set;

a terminator configured to determine, based on a criteria, whether to modify the execution of the software application using the second parameter set and repeat operations of the extractor, data store and fitness determiner for the second data set; and

a reporter configured to generate tuning results based on the parameters, the log values and the goal.

13. The system of claim 12, further comprising a validator configured to validate the log values after receipt by the extractor.

14. The system of claim 12, wherein the fitness determiner is further configured to score the first parameter set based on the received log values, a goal, and a log value from the data store.

15. The system of claim 12, wherein the hypothesizer is further configured to combine the first parameter set with the selected parameter set to produce a second parameter set based on a genetic algorithm.

16. The system of claim 12, wherein the terminator is further configured to have the criteria met when all appropriate parameter values have been used to modify the execution of the software application.

17. The system of claim 12, wherein the terminator is further configured to have the criteria met when, over a period of time, the incremental improvement resulting from applied parameter sets does not exceed a predetermined threshold.

18. The system of claim 12, wherein a parameter can include a combination of application settings.

19. The system of claim 12, wherein the software application is a virtual machine.

20. The system of claim 12, wherein the goal is reducing application latency.

21. The system of claim 12, wherein the hypothesizer is further configured to select the parameter to be retrieved from the data based on a score of the stored parameter.

22. The system of claim 12, wherein the fitness determiner is further configured to use a supervised or unsupervised learning technique to score each parameter set.

23. A computer-readable medium having computer-executable instructions stored thereon that, when executed by a computing device, cause the computing device to perform a method of automatically tuning a software application, comprising:

a) modifying execution of the software application using a first parameter;

b) scoring the first parameter based on a log value from the software application and an improvement goal;

c) storing the first parameter, the score of the first parameter and the log value in a data store;

d) combining the first parameter with a selected parameter retrieved from the data store, the combining resulting in a second parameter;

e) repeating (a) through (d) until a criteria is met; and

f) when the criteria is met, generating tuning results based on the parameters, the log values and the goal.

CLAIMS:

1. A method of automatically tuning a software application (150), comprising:

a) modifying execution of the software application (150) using a first parameter (121), said first parameter (121) being a parameter (121) able to change the operation of said software application (150);

b) scoring the first parameter (121) based on a log value (155) from the software application (150) indicating a result of said execution of said software application (150) and multiple ranked improvement goals (330, 340, 350), said multiple improvement goals (330, 340, 350) being associated with execution of said software application (150) and being ranked in order of importance, and said scoring being configurable and specific to said software application(150) and said improvement goals (330, 340, 350);

c) storing the first parameter (121), the score of the first parameter (121) and the log value (155) in a data store;

d) combining the first parameter (121) with a selected parameter (121) retrieved from the data store, the combining resulting in a combined parameter (121);

e) repeating (a) through (d) until a criteria is met, said criteria comprising that all parameters have been tried; and

f) when the criteria is met, generating tuning results based on the parameters (121), the log values (155) and the ranking of the multiple improvement goals (330, 340, 350), by selecting a first parameter having a highest scoring as a tuning result,

wherein one of said multiple improvement goals is reducing pause time.

2. The method of claim 1, further comprising, validating the log value (155) before said step of scoring the first parameter (121), wherein said validation of said log value (155) comprises determining if the log value (155) can be used for said scoring.

3. The method of claim 1, wherein said step of scoring the first parameter (121) comprises scoring the first parameter (121) based on a log value (155) from the software application (150), a log value (155) from the data store, and the improvement goal.

4. The method of claim 1, wherein the combining is performed using a genetic algorithm.

5. The method of claim 1, wherein the criteria is met when, once or over a period of time, the improvement in the log value compared to a previous log value indicating a result of a previous execution of said software application (150) does not exceed a predetermined threshold.

6. The method of claim 1, wherein a parameter (121) can include a combination of application settings.

7. The method of claim 1, wherein the log value (155) is one of pause time, throughput or memory footprint.

8. The method of claim 1, wherein the multiple ranked improvement goals (330, 340, 350) for reducing pause time is ranked as most important of the improvement goals.

9. The method of claim 1, wherein the selected parameter (121) retrieved from the data store is selected based on a score of the parameter (121).

10. A system to automatically tune a software application (150), comprising:

an extractor (210) configured to receive log values (155) from an operation of the software application (150), the operation of the software application (150) being affected by a first parameter set;

a data store configured to store the received log values (155) and the first parameter set;

a fitness determiner (320), configured to score the first parameter set based on the received log values (155) and multiple ranked improvement goals (330, 340, 350);

a hypothesizer (230) configured to retrieve a selected parameter set from the data store and combine the first parameter set with the selected parameter set to produce a combined parameter set;

a terminator (240) configured to determine, based on a criteria, whether to modify the execution of the software application (150) using the combined parameter set and repeat operations of the extractor, data store and fitness determiner for the second data set; and

a reporter configured to generate tuning results based on the parameters, the log values (155) and the multiple ranked improvement goals (330, 340, 350).

11. The system of claim 10, further comprising a validator (220) configured to validate the log values (155) after receipt by the extractor.

12. The system of claim 10, wherein the fitness determiner is further configured to score the first parameter set based on the received log values (155), a goal, and a log value (155) from the data store, or wherein the fitness determiner is further configured to use a supervised or unsupervised learning technique to score each parameter set.

13. The system of claim 10, wherein the hypothesizer (230) is further configured to combine the first parameter set with the selected parameter set to produce a combined parameter set based on a genetic algorithm; or wherein the hypothesizer (230) is further configured to select the parameter to be retrieved from the data based on a score of the stored parameter.

14. The system of claim 10, wherein the terminator is further configured to have the criteria met when all appropriate parameter values have been used to modify the execution of the software application (150); or wherein the terminator is further configured to have the criteria met when, over a period of time, the incremental improvement resulting from applied parameter sets does not exceed a predetermined threshold.

15. The system of claim 10, wherein the software application (150) is a virtual machine.

16. A computer-readable medium having computer-executable instructions stored thereon that, when executed by a computing device, cause the computing device to perform a method of automatically tuning a software application (150), comprising:

a) modifying execution of the software application (150) using a first parameter (121) , said first parameter (121) being a parameter (121) able to change the operation of said software application (150);

b) scoring the first parameter (121) based on a log value (155) indicating a result of said execution of said software application (150) from the software application (150) and multiple ranked improvement goals (330, 340, 350), said multiple ranked improvement goals (330, 340, 350) being associated with execution of said software application (150) and ranked in order of importance, and said scoring being configurable and specific to said software application and said improvement goal;

c) storing the first parameter (121), the score of the first parameter (121) and the log value (155) in a data store;

d) combining the first parameter (121) with a selected parameter (121) retrieved from the data store, the combining resulting in a combined parameter (121);

e) repeating (a) through (d) until a criteria is met, said criteria comprising that all parameters have been tried; and

f) when the criteria is met, generating tuning results based on the parameters (121), the log values (155) and the ranking of the multiple improvement goals (330, 340, 350), by selecting a first parameter having a highest scoring as a tuning result,

wherein one of said multiple improvement goals is reducing pause time.