

## A Mechanism for Reducing Size and Securing Data in Test and Measurement Archives

a report by

**Dr Hassan A Artail**

*Assistant Professor, Department of Electrical and Computer Engineering, American University of Beirut*

With the availability and affordability of measurement technology that is capable of delivering data with high resolution and recording frequency, scientists and engineers are faced with data archives that are growing exponentially in size. In test and measurement applications, it is not unusual to deal with files that are tens or even hundreds of megabytes in size, which has made the task of finding, processing and viewing relevant information within the data very time-consuming and non-trivial. Furthermore, the size of such data files presents major challenges relative to archiving, especially in environments that constantly output data and require data retention for legal or business purposes. One class of such applications is concerned with automotive testing and, in particular, car emissions testing. In this case, a prototype car is tested repeatedly (in the range of 30 to 50 times) and output data files are 5MB in size on average. These files contain multiple time series (channels) that describe the behaviour of hardware, systems and processes over a given period of time. They must be retained for at least seven years as stipulated by the Environmental Protection Agency (EPA) and for backtracking car history.

### Superfluous Data

In such applications it is customary for engineers (or scientists) to configure the test environment for collecting data from channels that are not necessary for the test and for the investigation of the particular problem. This is due to the complex interactions between the test object parameters (e.g. car subsystems) and the fact that the channels that would possibly

contribute to the expected phenomenon are not always known to the test engineers. Faced with this, engineers tend to acquire data from channels that end up not playing any role in the condition that is being investigated. As a result, and in many situations, much of the data collected is not critical or even useful for drawing case-specific conclusions and is only acquired because such knowledge was lacking. In all, such superfluous data has minimum or no archival value and therefore can be left out. Unfortunately, however, system administrators do not have a clue as to what part of the data this situation applies to.

From this, it becomes obvious that a post-processing algorithm that is able to detect data channels that are correlated or stand-alone is best suited as a solution to the problem. The type of correlation that should be searched for may be application-specific, but the concept remains the same – detect and keep data channels that are bound by some given interactions and discard the remaining ones while accounting for special cases and conditions (rules) that may be considered exceptions.

### Shrinking File Size by Discarding Data

Most of the techniques that were proposed in the literature to work with large data sets were meant to make the task of mining and searching within data simpler and more effective. The focus was on providing fast indexing techniques,<sup>1,2</sup> describing the data content,<sup>3</sup> discovering patterns and surprises in time series<sup>4-7</sup> and improving the efficiency of multilevel trend and surprise queries on time-sequence data.<sup>8</sup> One



Dr Hassan A Artail is Assistant Professor in the Department of Electrical and Computer Engineering at the American University of Beirut (AUB). His research is in the areas of Internet and mobile computing, distributed systems, data mining and knowledge management, in addition to computer and network security. He worked as a system development supervisor at the Scientific Labs of DaimlerChrysler, Michigan before joining AUB in 2001. At DaimlerChrysler, he worked for 11 years in the field of software and system development for vehicle testing applications, covering the areas of instrument control, computer networking, distributed computing, data acquisition, and data processing. He obtained a PhD from Wayne State University in 1999.

1. Yi B, Faloutsos C, "Fast Time Sequence Indexing for Arbitrary Lp Norms", In: *Proceedings of the VLDB Conference, Cairo, Egypt (2002)*.
2. Keogh E, "A tutorial on indexing and mining time-series data", *Proceedings of the IEEE International Conference on Data Mining, San Jose, CA (2001)*.
3. Murthy S, "Automatic construction of decision trees from data: a multi-disciplinary survey", *Data Mining and Knowledge Discovery (1998)*, 2 (4): pp. 345–389.
4. Bettini C, Wang S, Jajodia S, Lin J, "Discovering frequent event patterns with multiple granularities in time sequences", *IEEE Transactions on Knowledge and Data Engineering (1998)*, 10 (2): pp. 222–237.
5. Chau T, Wong A, "Pattern discovery by residual analysis and recursive partitioning", *IEEE Transactions on Knowledge and Data Engineering (1999)*, 11 (6): pp. 833–852.
6. Fawcett T, Provost F, "Activity monitoring: Noticing interesting changes in behavior", In: *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, San Diego, (1999), CA, pp. 53–62*.

Figure 1: Algorithm Components Used in Discovery of Index Information in Test and Measurement Files

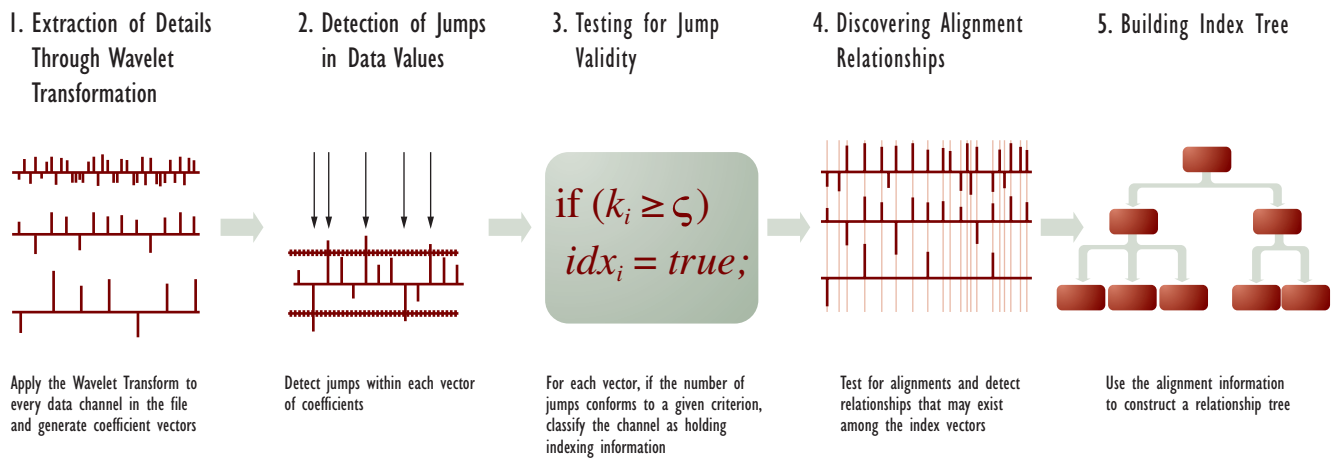
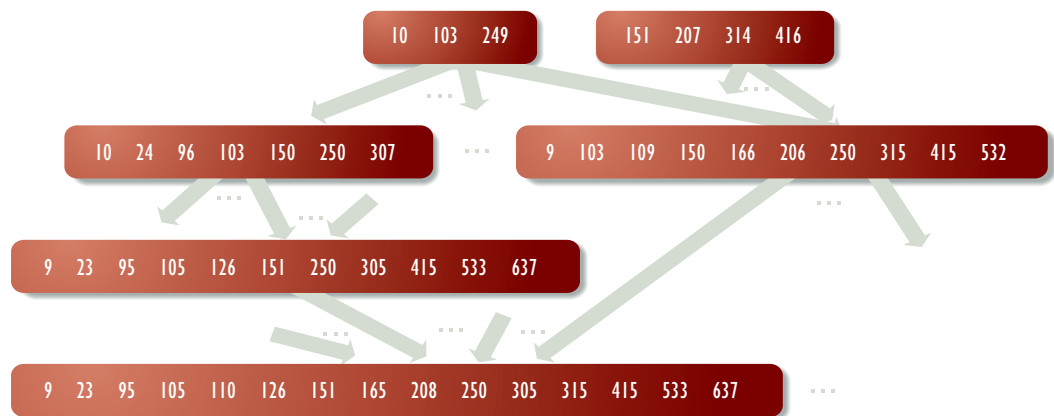


Figure 2: An Example of a HIL-tree with Two Root Nodes



approach, which is termed the HIL-tree algorithm, was proposed in the literature very recently and works by detecting correlating channels.<sup>9</sup> Actually, the HIL-tree is a data structure of index vectors that get constructed by an algorithm that looks for events and changes in trends within the raw data and then searches for alignments among such events across channels to determine relationships. The nodes of the tree store location information (time positions) about those events and thus serve purposes other than relating channels to each other. A general view of the algorithm is depicted in Figure 1.

In more detail, the algorithm uses the wavelet coefficients of the data in the file to check for jumps within each channel by examining neighbouring coefficients. Next, the detected jumps are tested against

a criterion to ensure that the total number of jumps does not exceed a certain fraction of the total number of samples in the channel. The rationale behind this test is to distinguish between meaningful events and normal fluctuations in the data. After recording the time positions of the jumps that pass the previous test in special vectors (one for each channel), an alignment check is performed across all these vectors for the purpose of discovering relationships among corresponding data channels. If all the time-position values of a vector match a subset of the values of another vector, or they are in very close proximity to them, then a relationship between the corresponding channels exists. For a pair of such channels, the one with fewer time positions is considered a parent of the other one. Figure 2 presents an example of a HIL-tree with index vectors holding time-position information

7. Keogh E, Leonardi S, Chiu W, "Finding surprising patterns in a time series database in linear time and space", In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Alberta, Canada, (2002)*, pp. 550-556.
8. Shahabi C, Tian X, Zhao W, "TSA-tree: A Wavelet-Based Approach to Improve the Efficiency of Multi-Level Surprise and Trend Queries", In: *Proceedings of the Scientific and Statistical Database Management Conference, Berlin, Germany (2000)*.
9. Artail H, "HIL-tree: A hierarchical structure for guiding search into test and measurement data archives", *Data Mining and Knowledge Discovery (2005)*, (10:3), pp. 229-250.

derived from data channels. This tree is an example output of the above-described algorithm and illustrates the type of relationships between the various data channels in the data file. The type of cause-effect relationship is not revealed, nor is it necessary for identifying interactions among data channels and, thus, the HIL-tree serves the purpose of isolating stand-alone channels that have no archival value.

**Car Emission Testing Data Files**

Although the storage space savings heavily depend on the type of application and the skills of the engineers or scientists who configure the test environment, an example is described to give an idea of the potential benefits of techniques such as the HIL-tree. The algorithm was applied against data files obtained from the car emissions test laboratories at a major automotive company.

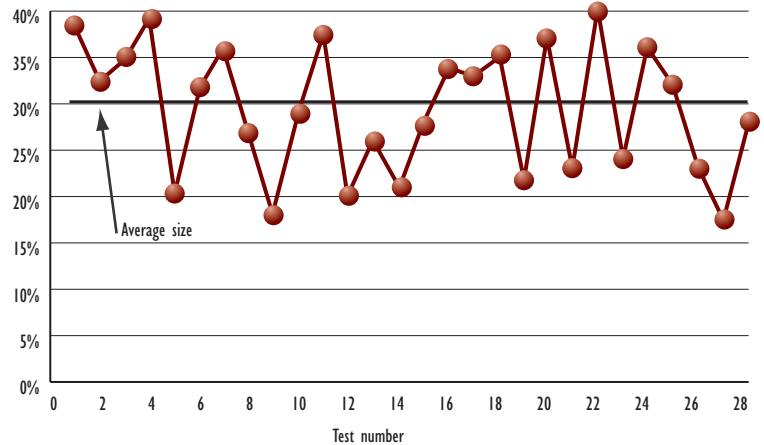
During their development stage, cars are tested repeatedly to ensure their compliance with governmental emission-level standards. They are tested inside test chambers on a dynamometer that practically generates real-world driving conditions. Data collected during the testing process is employed to fine-tune the car’s emission controls without compromising drivability and other performance criteria. Engineers collect all types of data that could potentially assist in analysing the performance of the vehicle and help identify root causes of behaviours and trends. In addition to measuring emission contaminant levels from the exhaust manifold and tailpipe, it is customary to collect channel values for ambient temperature and humidity, barometric pressure, target and actual speeds, light intensity, tens of engine-related variables (from the engine controller), plus many other channels that describe the test procedure, the driver and the test equipment. Moreover, data can be collected at various and varying rates that fall in the range of one to 500Hz. The vehicle engineer specifies the rate and channels to acquire during testing using an electronic test request, which gets submitted and scheduled to the test chamber system. For the majority of tests, data is collected at 1Hz and the average size of such files is 3.5MB.

In such data files, an example of a channel that exhibits changes in trends is the speed of the vehicle, which may steadily increase for a period of time (acceleration), decrease (deceleration) or stay constant (cruise or idle). Events, on the other hand, occur in channels that mostly describe behaviours of on/off-type devices and devices that exhibit constant-value behaviours, such as gearshifts.

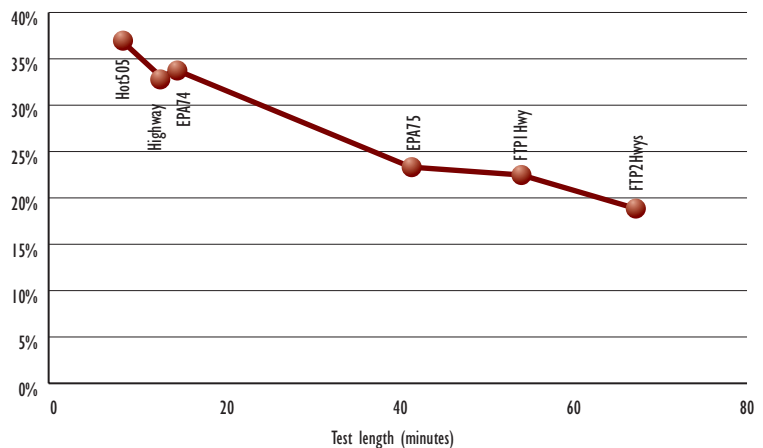
**Data Warehousing**

Emission testing is performed ultimately to comply with governmental regulations, namely those of the

*Figure 3: The Relative Size of 28 Files Output by the HIL-tree Algorithm – The Relative Size is Equal to the Size of the New File Divided by that of the Original File*



*Figure 4: Average Relative Size Per Test Type Versus Test Length*



EPA, which puts strict limits on the levels of emissions that a new car model can emit. Every car manufacturer must test every new car model and present the passing test results to the EPA before the car is introduced on the market. Furthermore, the EPA requires that the manufacturer keep a record of the results of all tests performed on the car for at least seven years. As the EPA’s standards are becoming increasingly stringent, car engineers are finding it necessary to collect more data, i.e. acquiring data from more channels and increasing the rate of acquisition. Besides the mandates of EPA, car engineers find it useful to tap into old test data for the purpose of learning successful testing strategies.

The above requirements can make it very challenging for system administrators to meet the increasingly growing storage requirements and satisfy engineers’ demands for fast retrieval systems. By noting that the EPA is only concerned about summary data that describe total emissions, vehicle configuration and environmental conditions (which occupy less than 1% of the total test archive space), it was clear that an opportunity existed for saving storage space and increasing data retrieval speed. Only if there was a way to identify the parts

of the test data that are deemed critical and relevant, and only archive these parts and what the EPA requires, could major storage space savings be realised. Given that the HIL-Tree data structure has the capability to do just this, it was applied to 28 real emission test data files for the purpose of measuring space savings. *Figure 3* shows the relative size of the files that were processed by the HIL-tree algorithm. As shown, the average space saving is more than 70%, which represents a major reduction in storage requirements and will likely have a positive impact on the speed of data retrieval from the data warehouse.

Upon inspection of the sizes of the new files, a relationship appeared to exist between the test type and the reduction in size. To demonstrate this, the relative sizes of all tests of the same type were averaged and the averages were plotted versus the length of each type (all tests of the same type have the same time duration). The results in *Figure 4* show that the percentage in size reduction increases as the length of the test increases. One explanation behind this relationship is that perhaps engineers requested more channels for longer tests. The shown labels indicate test types that correspond to different durations and different speed-versus-time profiles.

In conclusion, tools such as the HIL-tree, can be used to counteract the trend of increased required

test data storage space, help system administrators double the storage capacity of their existing data warehouses and, as a 'side benefit', provide users with a faster data retrieval experience.

### Data Security

The HIL-tree is a separate structure from the data file and therefore can be stored in a different physical storage location. The entries in the nodes of the tree are integer values that signify time information, which indicates positions of events in the channels of the data file. It follows that the values in the nodes can be used as elements of encryption keys that could be employed to encrypt the data in the corresponding channels. Since each node in the tree corresponds to a channel in the file, a capability exists for encrypting each channel with a different key (for added security) or selectively encrypting certain channels that are considered to hold sensitive data and thus guard it against intrusions.

The above-described encryption technique is very resistant to attempts to uncover the encryption key because the employed keys are not generated by a pseudo-random number generator but, rather, they are retrieved from the data itself through the HIL-tree algorithm. The degree of security in this case becomes a subject of the extent to which the HIL-tree information is secured. ■