Robotica (2000) volume 18, pp. 273–279. Printed in the United Kingdom © 2000 Cambridge University Press

A value-based scheduling approach for real-time autonomous vehicle control

D. Prasad and A. Burns

Real-Time Systems Research Group, Department of Computer Science, University of York, York YO10 5DD (UK)

SUMMARY
In future real-time systems such as those required for intelligent autonomous vehicle control, we need flexibility in choosing the set of services to support under varying environmental conditions and system states. It is not feasible to make an optimal choice of services at run-time, so we propose a method of ranking the services pre-run-time, based on the 'utility' of each service. This paper focuses on the problem of calculating a 'value' for the utility of each service alternative. We show how to derive values systematically and rationally, using Measurement Theory and Decision Analysis. The approach relies on engineering judgement and data input by a domain expert. In the context of autonomous vehicles, we believe that such knowledge would be available, making 'value-based scheduling' a feasible approach.

KEYWORDS: Autonomous; Vehicle control; Real-time systems; Service utility.

1. INTRODUCTION

Current real-time systems are scheduled statically, or pre-run-time, with a fixed set of 'hard services' (those which cannot afford to miss any deadlines) being mapped onto the available resources. Pre-run-time checks are used to ensure that all timing constraints are satisfied. In contrast, future real-time systems such as those required for autonomous vehicle control will need more flexibility in the run-time phase, to accommodate dynamic information into their operation. This type of run-time flexibility is needed to enhance the dependability and performance of the system while ensuring a basic level of functionality in all circumstances.

The traditional static approach uses only well-known attributes of the services such as deadline, minimum inter-arrival time, and worst case execution time. To enable dynamic scheduling, there is a notion that has been receiving attention in the real-time systems literature, called the *utility* of a service. It is the intrinsic benefit obtained by running that particular service and can be represented by some measure called its *value.* Consider that at various decision points (at run-time), a set of services are available for execution. At each point, the current resources may be insufficient to satisfy all services. Hence, decisions must be made to identify the 'extra' services to support when resources are spare, or to select the services to sacrifice when resources are scarce. *Value-based scheduling* is this decision problem of choosing services to execute from a given collection. so that the 'best possible' outcome ensues.

Although the idea of using utility for achieving flexible behaviour has been extensively promoted in the real-time literature,[1–4] very little has been said about how to derive the values in the first place. This is surprising, as without knowing how to generate appropriate values (describing the perceived utility of a service under specific conditions), the very idea of value-based scheduling is meaningless. In this paper we investigate the questions "Where do the values come from?" and "How are the values to be assigned?". We present some answers to these questions and illustrate them through a paper example of a vehicle controller that has to function autonomously in a changing environment, without any intervention from a human operator.

The rest of the paper is organised as follows: Section 2 describes the architecture of the real-time system and its services, and some typical parameters that affect their values. Section 3 uses an example of an autonomous vehicle controller to instantiate the model. Section 4 describes the overheads involved in value-based scheduling, and Section 5 proposes the idea of using 'modes' for overcoming some of these overheads. Section 6 outlines the mathematical framework for computing values, based on measurement theory and decision theory. In Section 7, we put together all the elements of our value assignment approach and apply it to the autonomous vehicle example. Section 8 extends the approach for practical application, and Section 9 outlines some directions for further work. The last section summarises and concludes the paper.

2. COMPUTATIONAL MODEL

We assume a computational model in which a system is deemed to consist of a set of *services.* Each service may be realised by one or more *service alternatives*—typically, only one of these is needed for any particular invocation of the service (although more than one could be run to enhance utility). Services are either periodic or sporadic and hence there is a bound on the number of possible invocations within any time interval. The entire system is considered to run indefinitely, hence each service has an unbounded number of invocations.

Let S be the set of services:

$$S = \{S_1, S_2, \ldots, S_n\}$$

Each service is composed of one or more (say m) alternatives:

$$S_i = \{A_{i_1}, A_{i_2}, \ldots, A_{i_m}\}$$

Each alternative is said to have the following attributes defined (there may be others): Minimum inter-arrival time (T), Deadline (D), Computation time or worst case execution time (C) and Value (V).

It is usually the case that T is defined for each service, but schemes in which each periodic service alternative has a distinctive period are possible. The first three terms are assumed to be known pre-run-time. In contrast, V may be dependent upon a number of run-time factors. If a service alternative completes it will result in some intrinsic benefit to the environment of the computer system. In general, the value of any service depends upon a number of criteria including:

(i) The quality of data used for the value approximation (e.g. accuracy, precision and statistical significance of the data).
(ii) The time at which the service completes (i.e. too early, acceptably early, optimally delivered. acceptably late, too late).
(iii) The history of previous invocations of this service.
(iv) The condition of the environment (e.g. visibility and temperature).
(v) The state of the computer system (what other services are being provided, and the load on the system).
(vi) The importance of the service (an initial classification being fundamental or essential, and non-fundamental or optional services).
(vii) The probability of completion of the service.

We need a systematic procedure for deriving the value of any service using the above factors. Before pursuing this problem further, let us apply the computational model to the example system.

3. THE INTELLIGENT AUTONOMOUS VEHICLE CONTROLLER EXAMPLE

Future real-time systems for autonomous vehicle control will need to exhibit intelligent and adaptive behaviour[4] in order to function in a highly dynamic and non-deterministic environment, characterised by the unpredictable nature of other vehicles, obstructions, route information, weather and road conditions. Further, the consequences of system failure due to faulty software, hardware or sensors may be catastrophic.

To realise such complex systems, two potentially conflicting objectives must be met: first. safety-critical and mission-critical services must be guaranteed to provide results of a minimum acceptable quality and reliability, by their deadlines; second, the utility of the system as determined by criteria such as the frequency, timeliness, precision and confidence level of the results produced, must be maximised.

The functionality of an autonomous vehicle control system has a wide range of timing requirements. Within a time frame of 10–100 seconds, it must plan and re-plan routes to reach the chosen destination, optimising fuel and time usage and take account of information about traffic conditions. On a shorter time scale (approximately 1 sec), scene recognition. assessment of other vehicle movements and path planning are required to ensure that the vehicle can steer a safe course, within comfortable ride limits. and carry out manoeuvres such as overtaking or merging with other traffic. Within a still shorter time frame of perhaps 100 ms, collision prevention algorithms need to sample sensor data, detect potential collisions with obstructions or other vehicles and initiate avoidance or braking.

From examining a typical autonomous vehicle control system we identify eight subsystems (i.e. services) and a number of alternatives within each. These are shown in Table I.

4. OVERHEADS IN RUN-TIME DECISIONS

Run-time decisions about which combination of services to select are only needed because there may not be adequate resources to run all services and their multiple alternatives at all times. In ‘small’ platforms like autonomous vehicles, the on-board computing resources are finite and we believe that there will always be more processing or communications required than the resources allow. We treat this as a fundamental assumption:

Assumption 1. *Although during specification and design, trade-offs have to be made between conflicting (or at least not compatible) requirements, we assume that run-time value-based scheduling is concerned only with the allocation of limited resources.*

Since resources within a real-time system such as an autonomous vehicle controller are limited, and may fail, we have to either curtail the number and quality of services, or

Table I. Autonomous Vehicle Example Services and Alternatives

Service	Service Alternative
Collision Prevention	Infra-red beam deflection RADAR Short Range Communication Stereo Vision Vehicle Tracking
Braking Control	Basic braking Anti lock braking (ABS) Load sensitive ABS
Engine Control	Basic control Increased precise computation (1) Increased precise computation (2)
Lateral Control (input processing)	Magnetic markers Line Following
Lateral Control	Proportional Integral Derivative Frequency Shape Linear Quadratic Fuzzy Rule-Based Neuro Control
Path Finding	Basic data fusion Fuel optimisation Ride comfort Time optimisation
Route Planning	Middle level planning GPS global planning
Displays Control	Basic update rates Speed warnings Fuel consumption analysis

find some way of selecting the services dynamically. In job-shop scheduling where the planning period is relatively long, and deadlines are specified in terms of minutes, if not hours, a dynamic allocation of resources may be probably feasible. However, dynamically scheduling real-time systems with millisecond deadlines is complicated by the following assumptions (which are supported by references [2] and [3]).

Assumption 2. *The run-time evaluation of an accurate and comprehensive value parameter for all services (and alternatives within a service) is prohibitively time consuming.*

Assumption 3. *The computation, for an accurate assessment of the total available resources (before some deadline) is prohibitively time consuming.*

Assumption 4. *Choosing the optimal subset of available services so that value is maximised and resource usage is bounded to the known available level is prohibitively time consuming (in general it is NP-hard).*

To overcome these difficulties, we propose a framework for static, or off-line computation of values by modelling the criteria that determine value through pre-run-time calculations. In autonomous applications where communication with 'ground crew' or human operators is not feasible (because the acceptable response time may be just a few milliseconds), it is imperative that the assignment of values be carried out pre-run-time. The next section describes our approach for overcoming the overheads implied by **Assumptions 2–4**, to make value-based scheduling feasible.

5. MODES AND STATIC VALUES

Section 2 listed many criteria that can affect the value of a service. For the present, let us restrict the variability in a service alternative's value, to just three criteria:

- The condition of the environment (e.g. 'wet' or 'dry' road conditions; 'day' or 'night' visibility)
- The condition of the underlying computing resource ('normal', 'overloaded' etc.)
- The state of the application software (what exactly it is doing at any instant)

The last cause embraces the history of the alternative's execution and the current state (accepted, rejected, guaranteed, or completed) of other related services or alternatives. We now define a *mode* in terms of these three variables:

$$\text{Mode} \overset{\text{def}}{=} [Env_{\text{mode}}, Comp_{\text{mode}}, Sys_{\text{mode}}]$$

The potential complexity arising from a dynamic value function can then be controlled by making the following assumptions:

Assumption 5. *Within a mode a fixed set of services and alternatives are defined.*

Assumption 6. *Within a mode the value of each alternative is constant.*

The last assumption is the key step which allows pre-run-time computation of values, and is realistic enough to be useful in an engineering context. Increasing the number of modes allows more specific dynamic behaviours to be accommodated, although care must be taken to ensure that mode explosion does not occur.

6. FRAMEWORK FOR COMPUTING VALUES

The idea of modes from Section 5 only solves the first part of our original problem, i.e., restricting the variability of the values. The second part of the problem is yet to be solved, i.e., the actual *derivation* of the values. To compute service values, we first need some means of mathematically underpinning the notion of value. Binary relations, commonly found in set theory can be used to compare the utilities of different services.

Let 'More_Useful' be such a relation:

$$(S_1 \text{ More\_Useful } S_2) \Rightarrow (V(S_1) > V(S_2))$$

where V is the 'value function' that assigns a real number to a service. If V is defined for only some of the S_i then it is called a *partially ordered* value function.

If V is defined for *all* S_i then it is called an *ordinal* value function:

$$(S_1 \text{ More\_Useful } S_2) \Leftrightarrow (V(S_2) > V(S_2))$$

Two basic problems must be addressed in deriving values: first, knowing whether a value function exists that can correctly *represent* the decision maker's preferences expressed in the binary relations, and second, knowing how to *construct* such a value function in practice. These are known in the literature[5] as the *representation* and *construction* problems, respectively. There are theoretical results in Measurement Theory[6] known as representation theorems which give sufficient and necessary conditions that must be fulfilled by the preference relations (such as More_Useful defined above) for specific analytical forms of value functions to validly represent them. The construction problem has been solved by various practical procedures[7,8] that are based on the results of the representation theorems.

For ordinal measurement to take place (i.e. for the ordinal value function to be used) the sufficiency conditions in its representation theorem must be verified. In practice, these can be tested by a pair-wise comparison experiment. For each mode defined in Section 5, the domain expert decision maker can be asked to express the preference relation between every pair of service alternatives available for execution in that mode. If the expressed preferences obey the required conditions (ideally, the binary relation should be asymmetric and negatively transitive,[6] then an ordinal value function can be constructed to represent them.

There are many mathematical techniques that implement the above ideas and one such technique is AHP (Analytic Hierarchy Process)[9] which is also supported by a software tool called Expert Choice. AHP allows more than just ordinal preferences to be articulated: it uses cardinal preference judgements as its input for constructing a value function. The services are compared for the *extent* of their relative utilities so that a cardinal value function can be generated. In this way, a measure of the 'distance' between alternatives and the 'ratio' of their relative utilities can be expressed, apart from just their ordering. In the next Section we apply this technique to generate values for the service alternatives in the autonomous vehicle example.

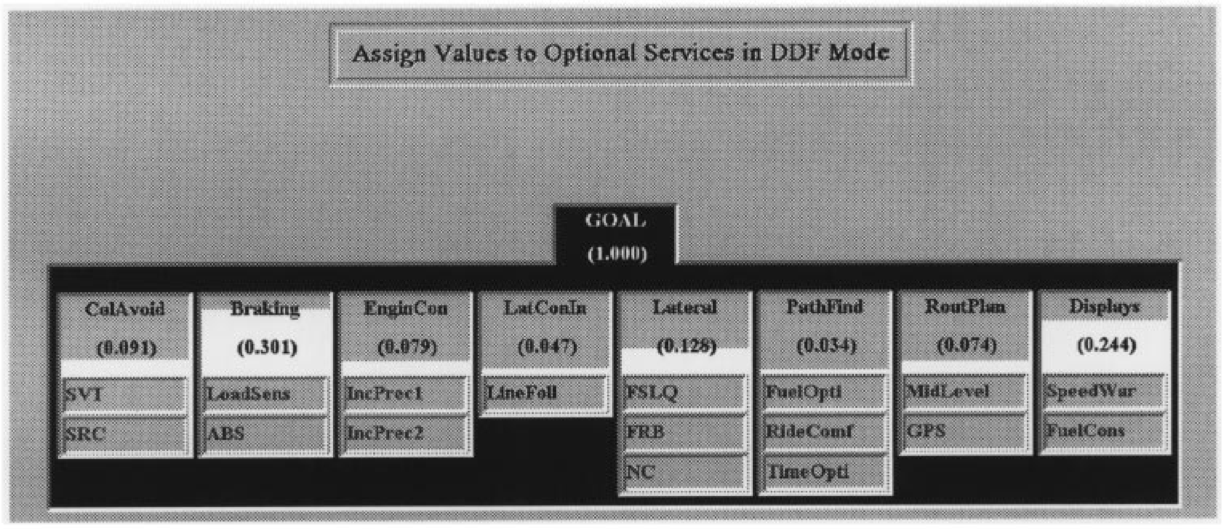
7. THE EXPERIMENT ON VALUE ASSIGNMENT

We show in this Section that the use of modes and a technique like AHP can help us to generate values systematically. If value is deemed to be constant within a mode (**Assumption 6**) then value assignment as discussed in this section can be an off-line activity.

7.1. *Modes in the example system*

An autonomous vehicle has a number of operational modes depending upon the road and traffic conditions (Env_{mode}), the state of the on-board computing resources ($Comp_{\text{mode}}$) and the performance of the services (Sys_{mode}) provided by the application. For illustrative purposes consider just two major modes representing two extreme scenarios. The first, *Day-time, Dry and Fault-free* (DDF) mode which refers to operation on a dry surface, in good visibility and with non-faulty hardware. The second mode *Night-time, Wet and Faulty* (NWF) refers to night time driving in wet conditions where some computing resources are malfunctioning.

In mode DDF we define two service groups: a *Mandatory* and an *Optional* group. The complete set of services and service alternatives for this example were given in Table I. Of these, we designate one or more alternatives under each service to be mandatory (and hence not subject to value-based scheduling) and the rest to be optional. Figure 1 shows the hierarchy of optional services and the glossary of abbreviations (compare with Table I). For the rest of this paper we focus our attention on just the DDF mode. A more extensive application of the value assignment approach can be found in reference [10].



Abbreviation	Definition
ABS	Anti Lock Braking
Braking	Braking Control Service
ColAvoid	Collision Avoidance Service
Displays	Displays Control Service
EnginCon	Engine Control Service
FRB	Fuzzy Rule based
FSLQ	Frequency-Shape Linear Quadratic
FuelCons	Fuel Consumption analysis
FuelOpti	Fuel Optimisation
GPS	GPS global planning
IncPrec1	Increased Precise 1
IncPrec2	Increased Precise 2
LatConIn	Lateral Control (input processing)
Lateral	Lateral Control Service
LineFoll	Line Following
LoadSens	Load sensitive ABS
MidLevel	Middle level planning
NC	Neuro Control
PathFind	Path Finding Service
RideComf	Ride Comfort
RoutPlan	Route Planning Service
SRC	Short Range Communication
SVT	Stereo Vision based Vehicle Tracking
SpeedWar	Speed Warnings
TimeOpti	Time Optimisation

Fig. 1. Optional services in DDF mode.

7.2. *The pair-wise comparison process*

Our objective is to obtain a value for each alternative that can be used for value-based scheduling during mode DDF. Given the set of services, the tool Expert Choice (EC) supports pair-wise comparisons and builds up an internal representation of preference judgements. If there are n services then a minimum of $(n-1)$ pair-wise comparisons must be made before an initial set of values can be generated, and a total of $n(n-1)/2$ comparisons would give complete coverage. Various graphical and tabular user-interface methods are supported by EC to enable these pair-wise comparisons to be entered.

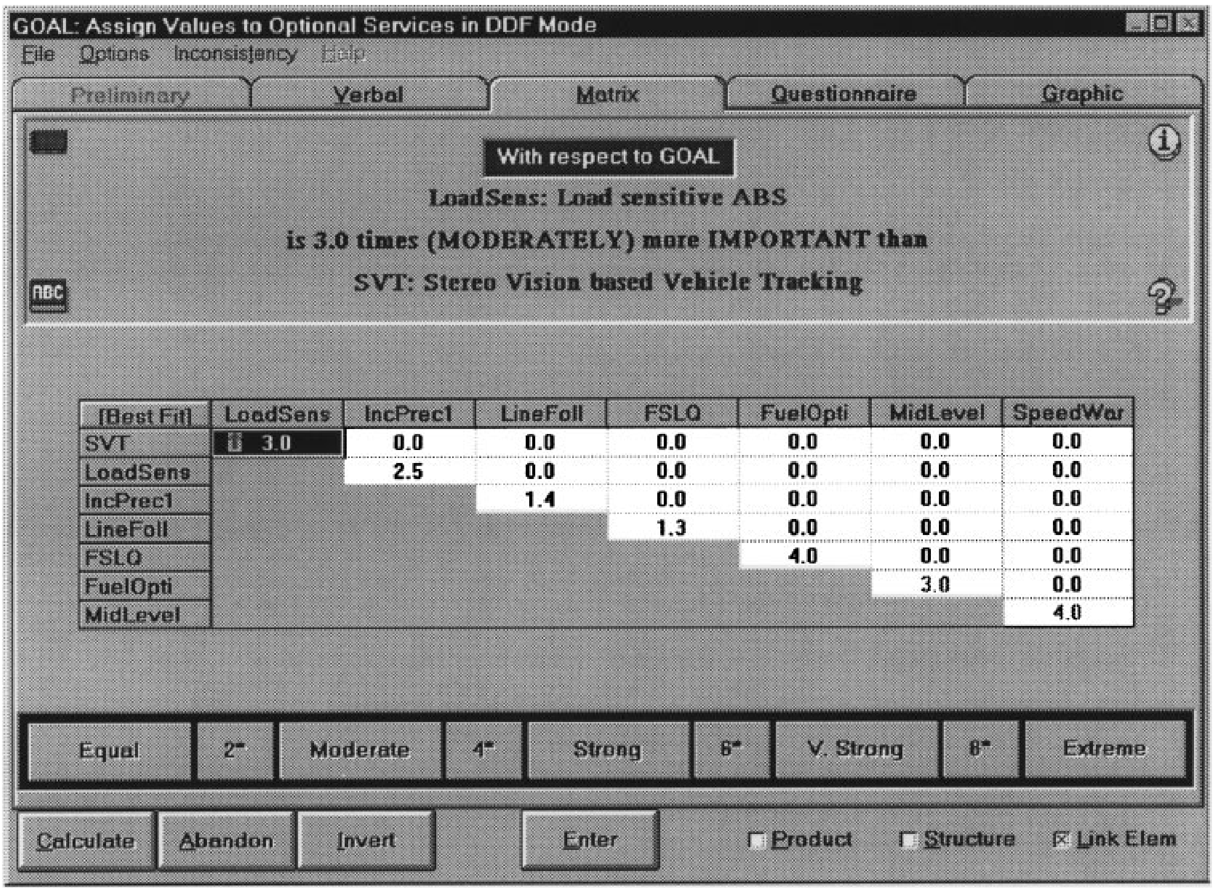
The comparison procedure is easy to carry out provided there is domain knowledge and engineering data available to provide the input required. In the context of autonomous vehicle applications, we believe that such knowledge would exist, based on which a domain expert could compare the alternatives within each mode. Figure 2 shows a screen dump of the *comparison matrix* with the 'best' alternative for each service represented in the rows and columns. The tool user (domain expert) fills in the diagonal entries and thus provides the minimum set of comparisons.

7.3. *The results*

Following data entry, EC produces two outputs: a string of real numbers (between 0 and 1) that represents the best fit of values to services; and an inconsistency index (ICI). This index gives an indication of the 'rationality' of the pair-wise entries. An ICI of 0 means complete consistency, while ICI close to 1 implies conflicting, inconsistent preferences. For example, with three services, A, B, and C, if A is twice as useful as B and B is twice as useful as C then A should be stated to be four times as useful as C.

Since Figure 2 involves only a minimal set of comparisons the inconsistency index is zero (i.e. the judgements are fully consistent). The output of the value assignment process, shown in Figure 3, is a set of values for the mode DDF. Once scaled to integers, they range from 183 for 'SpeedWar' (Speed Warning Display update), to 8 for 'TImeOpti' (Time Optimisation in Path Finding). These values (V) and other characteristics such as deadline (D), worst case computation time (C), and period (T) can then be used to carry out Value-based scheduling as has already been described in existing literature.

Eliciting pair-wise judgement from an expert in this way does not guarantee that 'real' preferences have been articulated, nor does it ensure that the 'real' utility of a service is captured by the value assigned to it. AHP only *infers* a set of values to numerically represent the input preferences. It is not possible ever to derive 'true' values since we do not know what they 'should be'. However, a full set of comparisons may arrive at values that are closer to the 'truth', although this may lead to higher levels of inconsistency. The AHP method can help the user reduce inconsistency by suggesting changes to specific judgements that would minimise the ICI. By trial and error iterations, a final, fairly consistent set of values can be generated.



With respect to GOAL

LoadSens: Load sensitive ABS

is 3.0 times (MODERATELY) more IMPORTANT than

SVT: Stereo Vision based Vehicle Tracking

[Best Fit]	LoadSens	IncPrec1	LineFoll	FSLO	FuelOpti	MidLevel	SpeedWar
SVT	3.0	0.0	0.0	0.0	0.0	0.0	0.0
LoadSens		2.5	0.0	0.0	0.0	0.0	0.0
IncPrec1			1.4	0.0	0.0	0.0	0.0
LineFoll				1.3	0.0	0.0	0.0
FSLO					4.0	0.0	0.0
FuelOpti						3.0	0.0
MidLevel							4.0

Fig. 2. Minimal set of pair-wise comparisons.

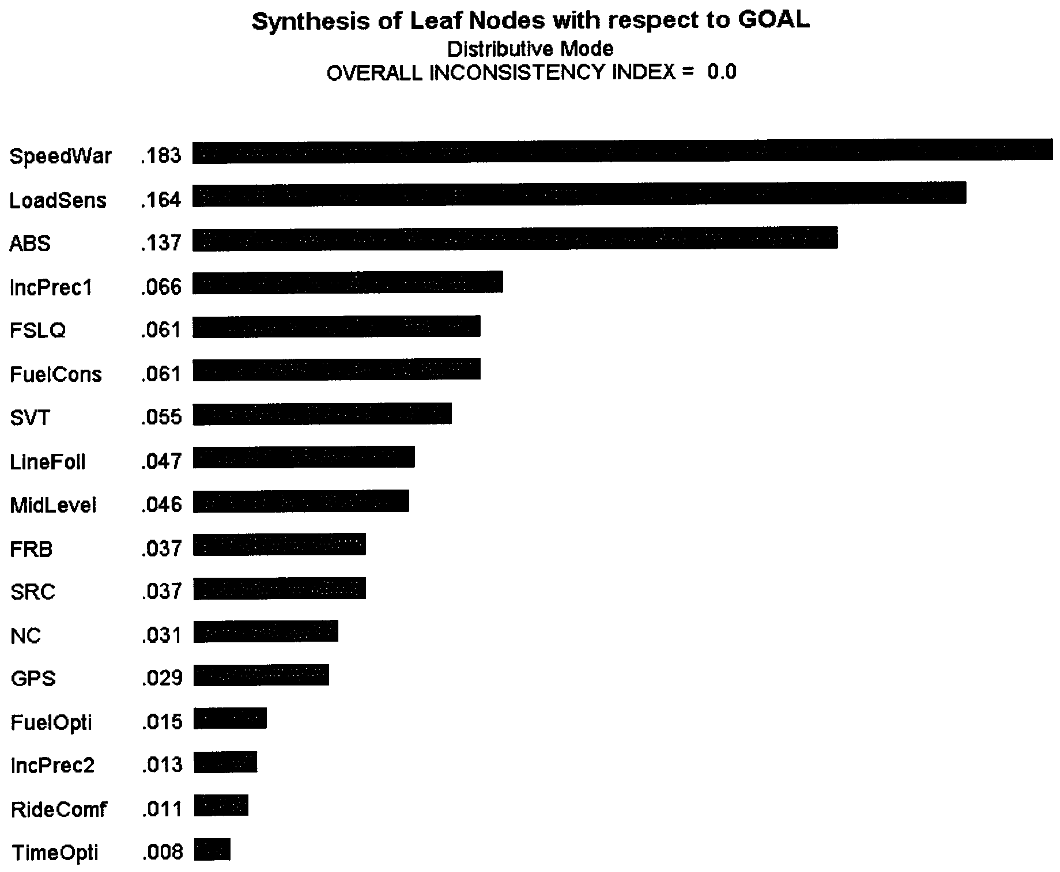


Fig. 3. Values derived from minimal judgements, with zero inconsistency ratio.

8. CONFIDENCE IN JUDGEMENTS

So far we have assumed that the decision-maker has equal confidence in all the input judgements that are entered. In practice, a domain expert would differ in the confidence placed in various judgements even if most of them are based on engineering data. We need a mechanism for incorporating a measure of confidence into the judgements (preferably in terms of probabilities) and using this in conjunction with the notion of inconsistency to arrive at the final set of values for the alternatives.

A pragmatic approach would be to elicit two matrices from the decision-maker: one of judgements, and the other of levels of confidence in the range [0, 1] for each judgement. The following scheme could then be used together with AHP's measure of inconsistency to create a 'best fit' matrix of pair-wise preferences that is used to generate the final set of values:

(i) Form the spanning set of judgements by selecting them in order of decreasing confidence levels. Since these are only a minimal set of judgements, there is no inconsistency yet.

(ii) Enter further judgements one-by-one in order of decreasing levels of confidence. while simultaneously reducing any resulting inconsistency by updating (changing) them using the following rule:

$$new = initial + (proposed - initial) * (1 - confidence)$$

Here, *initial* is the initial judgement entered by the decision-maker and *proposed* is the value proposed by AHP to reduce inconsistency.

In this way, the entire matrix can be filled by taking into account both confidence and consistency. Further investigation of this approach would require data based on 'real' examples, so that the results could be verified from experience of use.

9. FURTHER WORK

The authors are currently investigating the following directions of further research to expand upon the work reported in this paper:

- Defining modes and service alternatives for a real autonomous vehicle system and conducting the experiment on them, to evaluate the approach and to gain application experience.
- Choosing a subset of alternatives rather than individual ones, especially when the alternatives interact in some way. For example, given adequate spare resources, is it better to run services S_x and S_y or services S_x and S_z?
- Investigating time-varying value functions.
- Investigating multiple criteria value functions[8,11] to take into account more factors that contribute to value, and the role of modes to represent some of these criteria.

10. CONCLUSIONS

We have investigated the idea of value-based scheduling, and in particular, the problem of assignment or derivation of values of service alternatives for run-time decision making in adaptive real-time systems such as those required for intelligent antonomous vehicle control. Our approach consisted of the following steps:

- Using Measurement Theory to develop a framework for defining utilities and hence values of individual services.
- Distinguishing between various 'modes' of operation such that each service has a constant utility within a mode.
- Computing the value of each service per mode, prior to run-time. using expert judgement to compare different services through a technique such as AHP. The values thus generated can then be used in static scheduling analysis or for run-time decision making.
- Integrating the notions of consistency and confidence in judgements to provide a systematic, rational value-assignment procedure.

We have illustrated the ideas through a paper example of an autonomous vehicle controller which executes a range of services in a highly dynamic environment. The illustration in this paper is only theoretical, and needs to be supplemented with a real test platform. The practical suitability of the approach can only be evaluated using specific applications in an industrial context. The paper has also outlined some directions of future research.

Although there is considerable literature in the real-time community on flexible scheduling and adaptive systems, this work lacks an engineering context and well defined application domains. Autonomous vehicle control provides such a context. Vehicle control cannot be purely static as the environment is too non-deterministic. Efficient and robust approaches are needed if resources are to be used effectively and real-time requirements satisfied. This paper has attempted to bring together real-time scheduling work and and vehicle control. Both topics have much to gain from this synergy.

Acknowledgements
The research described in this paper is funded by Grant No. GR/L73456 of the UK Engineering and Physical Sciences Research Council.

References

1. C.D. Locke, "Best Effort Decision Making for Real-time Scheduling", *PhD Thesis* (Computer Science Department, CMU, CS-86-13, 1986).
2. J.W. Wendorf, "Implementation and evaluation of a time-driven scheduling processor", *Proceedings of IEEE Real-Time Systems Symposium* (1988) pp. 172–180.
3. H. Tokuda, J.W. Wendorf and H.Y. Wang, "Implementation of a time-driven scheduler for real-time operating systems", *Proceedings of IEEE Real-Time Systems Symposium* (December, 1987) pp. 271–280.
4. R. Davis, S. Punnekkat, N. Audsley and A. Burns, "Flexible scheduling for adaptable real-time systems", *IEEE Real-Time Technology and Applications Symposium* (May, 1995) pp. 230–239.
5. D.H. Krantz, R.D. Luce, P. Suppes and A. Tversky *Foundations of Measurement: Additive and Polynomial Representations*, **volume 1** (Academic Press, New York, 1971).
6. F.S. Roberts, *Measurement Theory, with Applications to Decision-making, Utility and the Social Sciences* (Addison Wesley, Reading, MA, 1979).
7. R.L.Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs* (Cambridge University Press, Cambridge, 1993).
8. P. Vincke, *Multicriteria Decision-Aid* (John Wiley and Sons, Chichester, 1992).,
9. T.L. Saaty, *Multiple Criteria Decision Making: The Analytic Hierarchy Process* (RWS Publications, Pittsburgh, PA, 1992).
10. A. Burns, D. Prasad, A. Bondavalli, F. Di. Giandomenico, K. Ramamritham, J. Stankovic and L. Strigini, "The meaning and role of value in scheduling flexible real-time systems", *J. Systems Architecture* (To appear).
11. D.K. Prasad, "Dependable Systems Integration Using Measurement Theory and Decision Analysis", *PhD Thesis* (Department of Computer Science, University of York, U.K., November, 1998).