Seeking Improved CPN Tools Simulator Performance:
Evaluation of Modelling Strategies for an Army Maintenance Process *

Guy Edward Gallasch[1], Benjamin Francis[2], and Jonathan Billington[1]

[1] Computer Systems Engineering Centre
University of South Australia
Mawson Lakes Campus, SA, 5095, AUSTRALIA
Email: guy.gallasch@unisa.edu.au jonathan.billington@unisa.edu.au
[2] Land Operations Division
Defence Science and Technology Organisation
P.O. Box 1500, Edinburgh, SA, 5111, AUSTRALIA
Email: benjamin.francis@dsto.defence.gov.au

Abstract. The availability of military equipment during a campaign depends on many factors including usage rates, spares, policy, and importantly the composition and distribution of maintenance personnel. This paper presents a Coloured Petri Net model of an Army maintenance process that encapsulates these factors. On simulating the model with CPN Tools we identify a number of simulation performance concerns. Using Standard ML profiling we discover that they are related to the modelling of personnel. Two different representations of personnel are created and evaluated in terms of simulation time using CPN Tools. We demonstrate that a list representation can have dramatic performance gains over a multiset representation. We further demonstrate that simulation performance can be improved by unfolding the CPN with respect to the maintenance network topology. Finally, we canvass some ideas for further improvements to simulation performance.
Keywords: Army Maintenance Process, Simulation Performance, Models of Personnel.

1 Introduction

In order to support both preparedness levels and the actual conduct of military operations, Defence is required to maintain the wide variety of defence equipment. This involves the deployment of tradespeople at a number of military workshops, which may be widely geographically distributed. The defence maintenance system can thus be considered to be a distributed system. Australia's Defence Science and Technology Organisation (DSTO) is interested in understanding the maintenance system and its processes at a deep level in order to suggest improvements to them. As part of a broader research initiative, DSTO have contracted the Computer Systems Engineering Centre (CSEC) of the University of South Australia to work on a collaborative project to model aspects of Defence logistics [8]. CSEC and DSTO are developing an Army Maintenance System Analysis Tool to examine the effectiveness of a deployed maintenance capability for the Australian Army. The tool is required to validate the feasibility of plans and to explore "what if" scenarios. Due to the distributed nature of the maintenance system, the tool is based on a timed Coloured Petri Net (CPN) [17, 19] model.

There have been several previous attempts to model and analyse Defence maintenance processes, e.g. [4, 15, 16, 22, 23, 26]. More generally, there are models of the maintenance of fleets of aircraft or vehicles [9], maintenance of power systems [10], models of specific maintenance facilities [27] and models that schedule maintenance activities to minimise the disruption caused to the normal operation of that equipment [2, 5], e.g. equipment in a production line, or military vehicles performing missions. There are also models that examine key equipment items from a maintenance system perspective [6, 25]. Some models [15,26] make use of the discrete event simulation tool ARENA [20], while others use Bayesian and Markovian analysis techniques [5, 9, 16] or optimisation of mixed-integer linear programming models [2]. A genetic algorithm is used in [23] to optimise maintenance schedules.

CPNs have a long history of being applied to the verification of complex distributed systems, e.g. see [1, 3]. In the area of Defence, CPNs have been applied to the modelling of Defence logistics networks [12,13], the modelling and analysis of operations planning and the subsequent production of a prototype tool based on CPNs [21,29], and modelling and analysis of the Australian Defence Force planning process [24].

* This work was supported by Australia's Defence Science and Technology Organisation, Contract No. 4500498737 and Research Agreement No. 229146.

Our first prototype timed CPN model of the Army Maintenance Process was presented in [14]. This model captured a number of aspects of Army maintenance, including equipment usage rates and the composition and disposition of maintenance personnel across a distributed network of maintenance workshops, and the impact of these aspects on the operational availability of a deployed system of equipment (the amount of time the equipment is up and running). The model allows the user to specify a distributed network of maintenance workshops, rather than a single maintenance facility, and is not specific to any single piece or type of equipment. Personnel are explicitly modelled and are not assumed to have homogeneous skills. Hence, our CPN model is more extensive, flexible and general than any we have encountered thus far in the literature. However, modelling with greater fidelity generally requires greater computational resources.

Although serving well as a *specification* model (a formal specification of the Army maintenance process), analysis was problematic. Attempts to simulate realistic scenarios with CPN Tools [7,18] failed due to excessive run-time. The investigations reported in [14] revealed that poor simulation performance was primarily related to the representation of maintenance personnel within the model. This paper expands on the investigations reported in [14] into the performance of the CPN Tools Timed CPN simulator in the context of this model. It provides a more thorough investigation of simulation performance of two models of personnel, and also considers an 'unfolded' alternative for the modelling of network topology.

The contribution of this paper is threefold. Firstly, this paper presents a overview of the refined version of the CPN model of the Army's maintenance process from [14]. Fusion places are no longer used, the use of code segments has been eliminated to a large extent, and guards have been expressed more concisely. In particular, we present a significantly revised version of the Assign Transport Resources page, which is key to our analysis. Secondly, based on the results of [14], we consider two data structures to represent personnel within our model and two ways to represent the topology of the maintenance network: encoding topology within tokens (folded approach) and representing topology explicitly within net structure (unfolded approach). Thirdly, we provide a more comprehensive comparison and evaluation of the performance of the CPN Tools simulator using these different personnel modelling approaches, and extend this investigation to consider the impact of an unfolded network topology on the performance of the simulator. Although our model represents a military maintenance process, the observations made may equally apply to other (similar) large-scale CPN models.

The rest of this paper is organised as follows. A brief introduction to the Army maintenance system is given in Section 2. Section 3 provides a description of our model. Choices for modelling personnel are described in Section 4. The simulation performance of these different models is presented in Section 5 and discussed in Section 6. Finally, Section 7 provides some concluding remarks. It is assumed that the reader is familiar with Coloured Petri Nets and CPN Tools.

2 Army Maintenance System

The Army maintenance system can be described at one level as a simple tree hierarchy of maintenance nodes arranged roughly along four distinct *lines of support*: from the 1^{st} Line (closest to the front line) where equipment is used in the field, up to the 4^{th} Line, where deeper level maintenance and equipment pooling occurs. Typically, the *National Support Base* for a military campaign exists at the 4th line of support. Such a maintenance system is illustrated in Fig. 1, where circles represent maintenance nodes and arcs represent the movement of personnel and equipment. Nodes within the network represent maintenance workshops. Each contains a range of personnel of different trade types, forward repair/recovery capabilities, and authorisations to conduct particular repairs. A maintenance node also has an inherent capacity that governs the amount and nature of work that can be accepted at a node. In addition, a node may simultaneously be a source of maintenance liability as they also operate equipment which may require maintenance. A *strategic interface* exists between 4th line and the other lines of support, symbolising that 4th line typically exists within the country of origin and is relatively static, whereas the remaining lines are deployed.

These node characteristics, combined with the various equipment failures, determines where items can be repaired in the system, and hence the extent to which equipment is moved around the maintenance network. An item of equipment that requires maintenance must be co-located with suitable maintenance facilities so that the maintenance can be performed. The equipment may already be at a suitable maintenance workshop, or it may be that a team of maintenance personnel can be dispatched from a nearby workshop and travel to the equipment. More often, however, this means the equipment needs to be transported to a maintenance workshop.

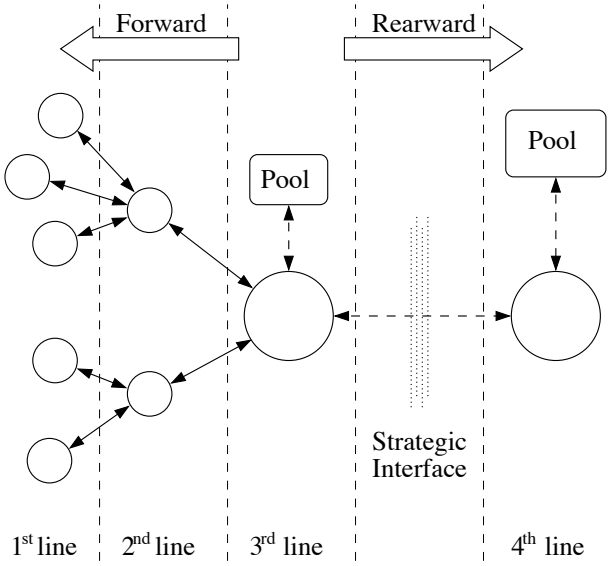

Fig. 1. Lines of Support within the Army Maintenance System.

In general, repair should be conducted as far forward as possible in order to reduce delays in returning equipment to an available state, thus increasing effectiveness. This may come at the cost of reduced efficiency due to additional delays in transporting personnel through the network. Alternatively, availability might be managed through releasing replacements from pools, which are in turn replenished with items that are repaired under lower operational tempo conditions.

The interaction between the degree of forward repair, the nature of repairs that can be conducted in the area of operations versus the National Support Base (i.e. 4th line) the disposition and type of personnel allocated to each line of support, and the use of replacement pools is a complex problem well suited to formal modelling and analysis. This allows force structure planners to holistically examine trade-offs involving the size and nature of the Army maintenance system through a review of performance over a wide range of operational scenarios. Typically, a realistic scenario will involve hundreds of personnel and thousands of pieces of equipment distributed over tens of nodes.

3 Model Description

Hierarchical Coloured Petri Nets are used to model the Army maintenance system. Figure 1 shows a *system*-oriented view of Army maintenance, with a geographically distributed set of nodes (workshops) with a given topology, where maintenance activities may be carried out at any of the nodes. Apart from the node at fourth line, the operations at each node follow a *maintenance process* that is very similar, regardless of the location of the node. This process differs only with respect to factors such as the capacity of the workshop, the grade of repair that can be carried out at the workshop, and the maintenance personnel located at the workshop. Because of these similarities, rather than taking a system-oriented view of Army maintenance in our CPN model, we have taken a *process*-oriented view. CPNs allow us to capture physical characteristics, such as network topology and individual node features, within its data structures, rather than within the net structure. We do so in our model, hence taking a process-oriented view avoids the need to duplicate the net structure relating to the maintenance process followed within each workshop. This approach eases maintenance of the model itself (e.g. in the event of a change to the maintenance process that we have captured, we need only implement the changes for one piece of net structure rather than a piece of net structure for every workshop). Our model is thus a hierarchical CPN model, representing a decomposition of the *processes* related to performing maintenance in a military environment.

The model consists of 14 pages arranged into three hierarchical levels, plus an additional layer required to initialise the model with a specific scenario. The hierarchical structure of pages within our model is illustrated in Fig. 2. The Process Overview and Workshop Maintenance pages serve as a flowchart of the

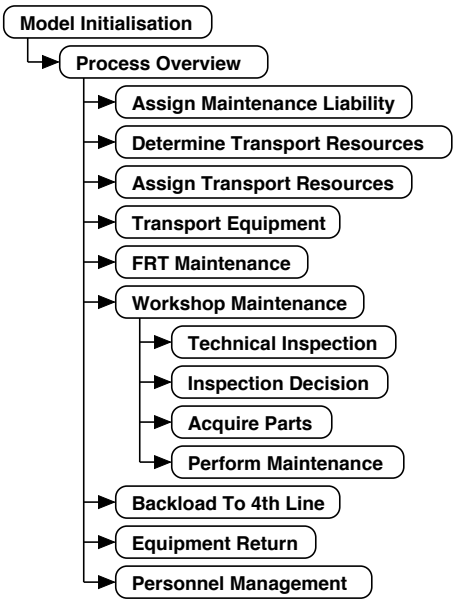

Fig. 2. The hierarchical structure of our CPN model.

process. We use these pages to describe the system and processes we are modelling. Size constraints prevent us from describing the model in detail in this paper. A full description is given in [11].

The model comprises 44 executable transitions and 14 *substitution* transitions. There are 27 distinct places in the model, with a total of 88 physical places due to port/socket place duplication. Further complexity is contained in extensive model inscriptions and function definitions, written in the programming language Standard ML [28], and comprising approximately 1600 lines of code.

Since it was first published [14], the model has been significantly improved. Originally, fusion places were used as part of the Model Initialisation page (see Section 3.2) to allow the initial marking of numerous places throughout the model to be given a scenario-specific pseudo-initial marking. Unfortunately, overcoming the limitation of CPN Tools not allowing a place to be both a fusion place and a socket place resulted in a mix of fusion places and port/socket places, which is undesirable from a hierarchical modelling perspective. The model no longer uses fusion places and hence the hierarchical structuring has been made clearer. Also, the previous model used code segments on a number of model pages to specify a binding of output arc variables. Where this was unnecessary, the code segments have been replaced by functions on output arcs. As will be seen in Section 4 we examine the use of two data structures for personnel within this model. The use of code segments has been eliminated completely from the first model (that considers each person as a separate token), an example of which is seen in Fig. 4 in Section 4.1. Code segments have only been retained in two instances in the second model (that considers people stored as values in lists) where pragmatic to do so.

3.1 Important Data Structures

Before we begin describing the model itself, we must first introduce three key data types used to describe the state information in the model. These are the `Equipment`, `Maintenance_Task` and `Personnel` colour sets. These three colour sets, combined in various ways, form the basis of the types of almost all places in the model.

Equipment The `Equipment` colour set is shown in lines 5-13 of Listing 1. It defines a record structure that describes individual items of equipment that will require maintenance. The equipment type, present location and home location (lines 5-7) are strings. The usage mode (line 8) designates whether an item of equipment is actively in use or is in an equipment pool (line 2). The last service type (line 9) records whether the last service was a major service or a minor service (line 3). Finally, the current usage meter reading, time of last service, usage meter reading at the last service and time of the last inspection (lines

Listing 1. The `Equipment` Colour Set.

```
1  colset Location = STRING;
2  colset Usage_Mode = with Active | Pool;
3  colset Service_Types = with Major | Minor;
4
5  colset Equipment = record equipment_type : STRING *
6                            present_location : Location *
7                            home_location : Location *
8                            usage_mode : Usage_Mode *
9                            last_service_type : Service_Types *
10                           usage_meter : INT *
11                           time_of_last_service : INT *
12                           usage_meter_at_last_service : INT *
13                           time_of_last_inspection : INT timed;
```

Listing 2. The `Maintenance_Task` Colour Set.

```
1  colset Maintenance_Type = with Service | Corrective | Inspection;
2  colset Grade = with L|M|H;
3  colset Trade = STRING;
4  colset ERT_by_Trade = INT;
5  colset Job = product Trade * ERT_by_Trade;
6  colset Job_List = list Job;
7  colset Priority = with Essential | Non_Essential;
8  colset Mobility = with wheeled_mobile | wheeled_not_mobile | not_wheeled;
9  colset Maint_Methods = with InSitu | SelfTransport | Distribution | Recovery | FRT;
10 colset Maint_Methods_Attempted = list Maint_Methods;
11
12 colset Maintenance_Task = record maintenance_type : Maintenance_Type *
13                                   next_occurrence_time : INT *
14                                   current_request_location : Location *
15                                   grade_required : Grade *
16                                   job_requirements : Job_List *
17                                   job_priority : Priority *
18                                   mobility : Mobility *
19                                   inspected : BOOL *
20                                   methods_attempted : Maint_Methods_Attempted *
21                                   assignment_timeout : INT timed;
```

10-13) are recorded as integers. Usage metrics could include e.g. distance travelled (odometer), operating hours, rounds fired, or calendar time.

Maintenance Tasks The `Maintenance_Task` colour set is shown in Listing 2. This is also a record (lines 12-21) that specifies the maintenance required and records the progression of individual items through the system. The type of maintenance required (line 12) can be either a regular service, corrective maintenance (in the event of a breakdown) or an inspection of a piece of equipment (line 1). The next occurrence time (line 13) records when the next 'future' maintenance event is due (unbeknownst to the system) to occur for a corresponding item of equipment. The location of the workshop that is currently considering the maintenance task is given by line 14. The grade of repair (line 15) required for a particular maintenance task will be either Light, Medium or Heavy (line 2) depending on the nature of the task, and hence will affect where and by whom the corresponding item of equipment can be maintained. The job requirements (line 16) specify the tradespeople required and the length of time for which they will be required (lines 5 and 6). Rather than using an enumerated type to specify the trade types, e.g. vehicle mechanic, our model uses strings (line 3). This is for extensibility and to overcome a CPN Tools limitation that prevents colour set declarations being specified externally and imported into the tool. The length of time a particular trade is required is given by the Estimated Repair Time (ERT) on line 4. The priority of each particular maintenance task (line 17) is categorised as either essential or non-essential (line 7) and is based on the type of equipment and the nature of the campaign. Line 18 specifies the mobility of the equipment to be maintained. According to line 8, the equipment can be either wheeled and mobile (e.g. a truck that can be driven), wheeled and not mobile (e.g. a truck with a broken engine) or not wheeled (e.g. a generator). Whether or not the equipment has undergone a technical inspection, revealing the current maintenance liability, is given on line 19. Line 20 specifies the methods of transport/maintenance that have been attempted in order to address this maintenance task. The methods are given on line 9 and will be described in Section 3.3. The assignment timeout (line 21) specifies how long a task will wait

Listing 3. The `Personnel` Colour Set.

```
1 colset Personnel_States =   with Ready | Working | Offline;
2
3 colset Personnel = record trade : Trade *
4                           home_location : Location *
5                           working_status : Personnel_States *
6                           last_came_online_time : INT timed;
7
8 colset Personnel_List = list Personnel timed;
```

for personnel to be assigned to it before some other alternative method of maintenance/transport is attempted.

Personnel Listing 3 describes the data structures used for personnel. `Personnel` (lines 3-6) records the trade (line 3), home location (line 4) and working status (line 5) of individual people. The working status specifies whether a person is ready to work, currently working, or *offline* (line 1). Offline means that the person is not currently available to be assigned to maintenance work (e.g. is sleeping or performing other duties). The time at which the person last came online (moved from Offline to Ready) is also recorded (line 6). Line 8 declares a list of personnel.

3.2 The Model Initialisation Page

The Model Initialisation page is used to populate the model with tokens representing the scenario under consideration. Because of our process-oriented modelling approach, no modification of the net structure is needed when analysing different scenarios. The Model Initialisation page initialises the following places:

- **Node Knowledge:** The Node Knowledge place is populated with tokens that describe the characteristics and capabilities of each maintenance workshop in the scenario;
- **Topology:** The Topology place specifies the topology of maintenance workshops (nodes);
- **Personnel:** The Personnel place is populated with tokens that represent the number and disposition of personnel to be distributed throughout the maintenance workshops;
- **Equipment Awaiting Maintenance Assignment:** This place is populated with all of the equipment in the system that will require maintenance;
- **Equipment Awaiting Parts and Equipment Ready for Maintenance:** As 'house-keeping', these places are populated with one token per maintenance workshop, allowing for a prioritised list of maintenance tasks to be stored in both places for each workshop, as will be described later.

This initialisation is achieved through the use of functions that read in initialisation data from text files on local storage. The text files may be populated by an external editor, e.g. from an Excel spreadsheet, thus allowing for increased modelling flexibility and expedient scenario generation and modification.

3.3 The Process Overview Page

The Process Overview page, shown in Fig. 3, is the highest level page that describes the maintenance process. The net structure of this page has been developed to explicitly represent the flow of equipment through the maintenance process, depicted by the bold arcs. This page consists mostly of substitution transitions, each of which represents a sub-process within the overall maintenance process. Each place on this page is typed by `Equipment` or by Cartesian products of `Equipment`, `Maintenance_Task` and `Personnel`. The names of these colour sets reflect their declarations (see [11]), e.g. `EquipmentXTask` is the product of `Equipment` and `Maintenance_Task`.

We now describe the processes represented on the Process Overview page.

Assign Maintenance Liability The starting point in the execution of our model is the assignment of a maintenance liability to each of the items of equipment in the Equipment Awaiting Liability Assignment place. This is done by the Assign Maintenance Liability page. Each item of equipment is paired with a 'future' liability (specific details of the next maintenance requirement, in the form of a maintenance task) as it is transferred to the Operational Equipment place, where it is considered to be in use until such

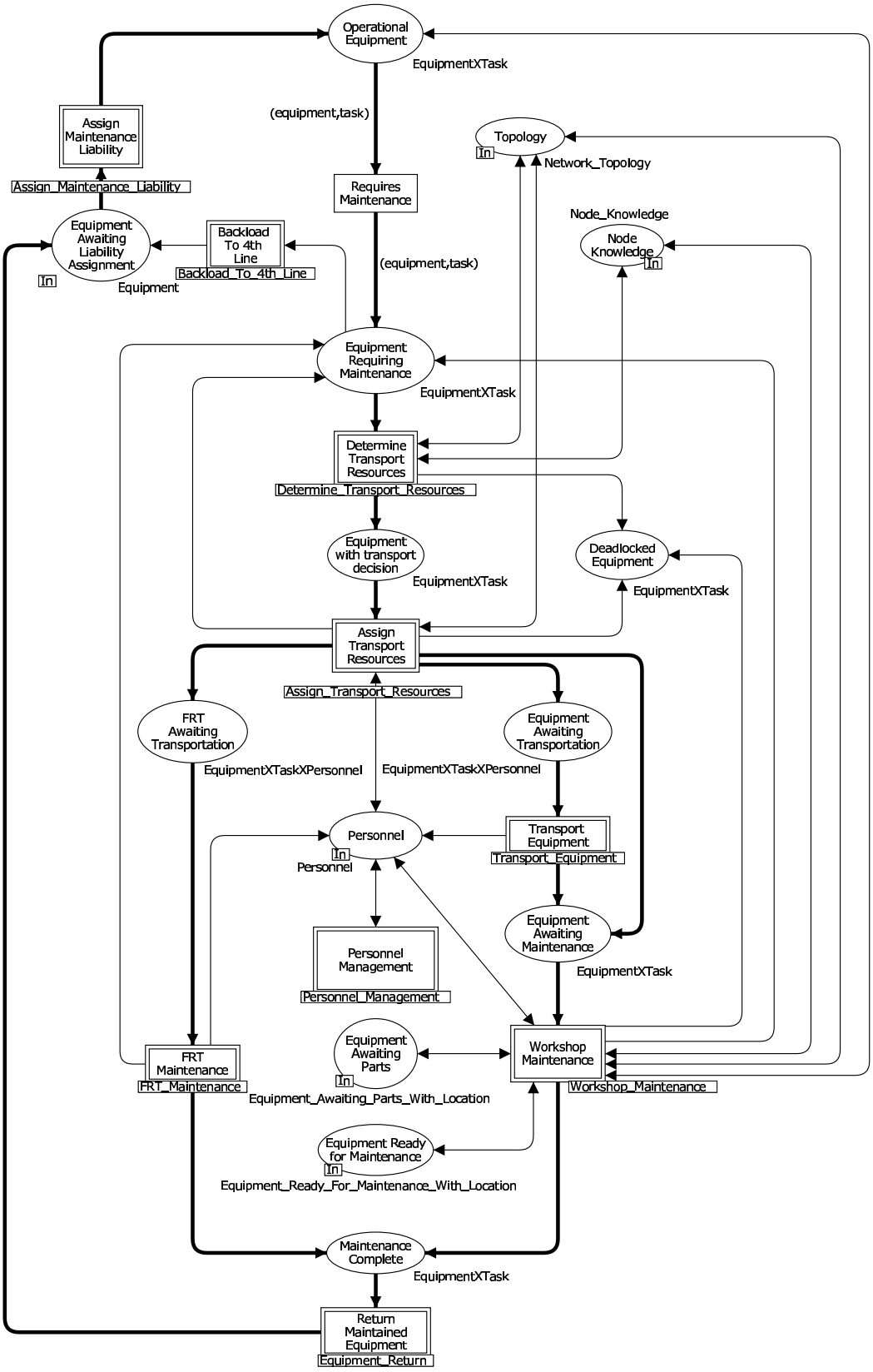

Fig. 3. The Process Overview page, showing the high-level structure and major flows of equipment (in bold).

time as its maintenance liability is realised. At this point, the Requires Maintenance transition moves the item of equipment from an operational state (Operational Equipment place) to a non-operational state (Equipment Requiring Maintenance place), where the item awaits maintenance. We use the timestamp mechanism of CPNs to control when this occurs.

Determine Transport Resources The first step in the maintenance process is to determine the transportation requirements. The approaches are:

- **In-situ:** the equipment does not need to move from its present location;
- **Self-transport:** the equipment moves itself to a suitable maintenance workshop;
- **Distribution:** the equipment is moved using a more general distribution network, e.g. road or railway, that services the demands of the military operation;
- **Recovery:** a team of personnel (a Recovery Team) moves from a maintenance workshop to the location of the equipment and brings the equipment back to the maintenance workshop; and
- **Forward Repair Team:** a team of personnel move from a maintenance workshop to the equipment location in order to effect the necessary maintenance.

The method is selected by the Determine Transport Resources subpage. All methods necessitate the use of a form of transportation with the exception of in-situ maintenance.

Assign Transport Resources Once the transport requirements have been identified, the Assign Transport Resources subpage ensures that the personnel requirements can be fulfilled and assigns the required personnel. In the event that the required personnel are not available, the equipment is passed back to the Determine Transport Resources page (the arc from Assign Transport Resources to Equipment Requiring Maintenance) and a different approach is selected. This process may be repeated a number of times both for the current location of the equipment and in the event that the equipment has been moved to a new location but that new location is not able to perform the necessary maintenance (hence the return arc from the Workshop Maintenance page to the Equipment Requiring Maintenance place). In the event that all options for transportation are exhausted and no further locations exist to which the equipment can be moved, this particular item of equipment is not able to be maintained by the system and is deposited into the Deadlocked Equipment place.

If the personnel requirements can be satisfied, there are three possible outcomes (the three bold arcs that exit the Assign Transport Resources page in Fig. 3). In the case of in-situ maintenance, no transportation is necessary, so the equipment item is passed directly to the Equipment Awaiting Maintenance place. Secondly, if a Forward Repair Team is required (defined by the type of item and location) the personnel comprising the FRT are passed to the FRT Awaiting Transportation place, along with the equipment and maintenance liability to which this FRT corresponds. Note that despite being present in the same token on the same place, the FRT and the item of equipment are still at geographically separate locations. Lastly, if an equipment item itself needs to travel within the maintenance network (self-transport, distribution or recovery), it is inserted into the the Equipment Awaiting Transportation place.

Transport Equipment An item of equipment that requires transportation may be able to transport itself, or it may require the distribution network or the the assistance of a Recovery Team to do so. All three of these scenarios are handled by the Transport Equipment page. Once all transportation is taken care of, equipment items will enter the Workshop Maintenance process from the Equipment Awaiting Maintenance place and any personnel required during the transportation will then be released to conduct other work.

FRT Maintenance and Workshop Maintenance Both the FRT and Workshop Maintenance processes result in maintained items of equipment. Workshop maintenance is described in Section 3.4. In the case of FRT Maintenance, the FRT travels to the equipment to maintain it, before returning to its assigned workshop and disbanding. However, it is possible that the FRT will fail to repair the equipment. This results in the equipment being reassessed by the Determine Transport Resources subpage and an alternative transportation approach chosen.

Backload to 4th Line In the case of either FRT or workshop maintenance, if an initial technical inspection reveals that an item of equipment is beyond the level of repair available, the equipment is 'strategically' recovered from the area of operation to the 4^{th} line of support. When this happens, a replacement item of equipment is issued from the 4^{th} line pool and inserted back into the area of operation at the location of the item it is replacing (after some transport delay). This process is denoted here as "Backload to 4th Line", and is modelled by the subpage of the same name. Like all equipment, the replacement must be given a "future" maintenance liability, hence the arc from Backload to 4th Line to Equipment Awaiting Liability Assignment.

Return Maintained Equipment Subject to the successful completion of the Workshop Maintenance or FRT Maintenance process, the equipment item is put into the Maintenance Complete place, and the maintenance personnel made available for another job, or moved into an off-line or resting state. The Return Maintained Equipment process will return the equipment to its original location. The equipment is inserted once more into the Equipment Awaiting Liability Assignment place, which permits the item to return to an operational state after having been assigned a new maintenance liability.

Personnel Management The Personnel Management page is responsible for moving personnel from a ready state to an offline state, and vice versa, at the correct times.

3.4 The Workshop Maintenance Page

The primary input to the Workshop Maintenance page is the Equipment Awaiting Maintenance place, containing equipment items (with their associated liability description) that are located at a maintenance workshop. The Workshop Maintenance page itself comprises four substitution transitions that describe four sequential sub-processes of workshop maintenance. A preliminary version of this page appears in [14]. These subpages are briefly described below:

Technical Inspection This process represents the inspection of equipment that reveals to the workshop the maintenance activities that must be undertaken for a particular item of equipment (the previously hidden maintenance liability).

Inspection Decision Firstly, this page determines whether to issue a replacement item of equipment to go into service while the other is undergoing maintenance. Secondly, it determines whether the current workshop can accept the new task (sufficient grade of repair, spare capacity and appropriately skilled personnel). Finally, accepted tasks and equipment are passed to the Acquire Parts process (below). If not accepted, the equipment is returned to the Equipment Requiring Maintenance place, so that it may undertake another iteration of the Determine Transport Resources process described in Section 3.3.

Acquire Parts The Acquire Parts process represents the delays inherent when spares are required but not presently in stock. Equipment items awaiting parts are stored in a queue prioritised by the expected arrival time of the parts.

Perform Maintenance Tradespeople are automatically assigned to the highest priority maintenance task at their location. Once assigned, the tradesperson continues to work on the repairs until they either complete their portion of the maintenance task (their job in the joblist of the maintenance task) or are required to rest. The completion of all required maintenance activities for a given item of equipment signals the end of the Perform Maintenance process, at which point the item of equipment is placed in the Maintenance Complete place, and the cycle begins again.

4 Modelling Personnel

Personnel enable transportation and maintenance activities. The Personnel place appears on 9 of the 15 model pages and the availability of tradespeople directly influences the enabling of 6 executable transitions across 4 model pages (Assign Transport Resources, Technical Inspection, Perform Maintenance and Personnel Management). The prominence of personnel within our model means that a large part of the computational effort of CPN Tools relates to the calculation of enabled binding elements involving personnel. The representation of personnel greatly influences simulation performance [14]. This section describes the two representations investigated in Section 5.

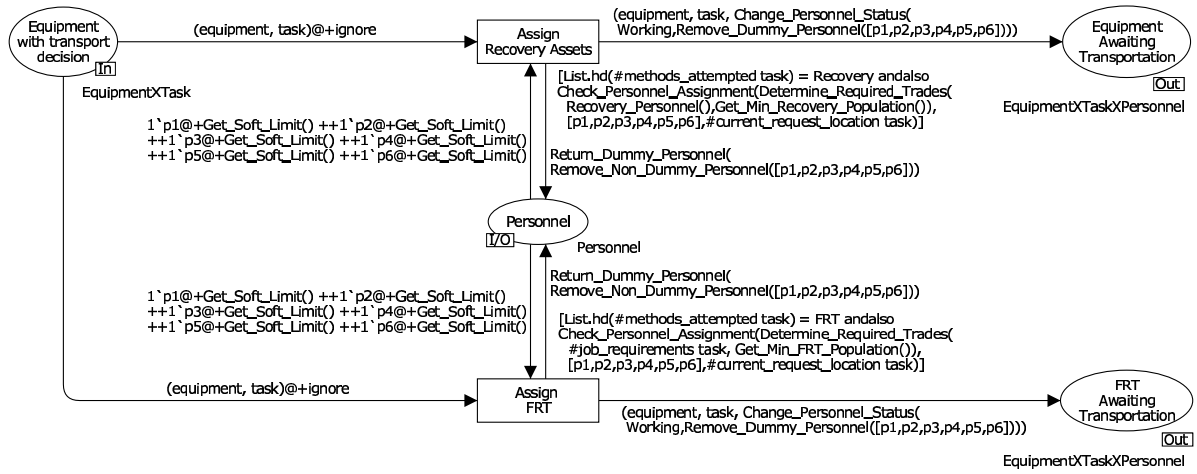

Fig. 4. A fragment of the Assign Transport Resources page of Model 1.

4.1 Personnel Modelled as Tokens (Model 1)

Our initial model represented personnel as individual tokens with timestamps, as per Listing 3. However, two difficulties were encountered: the number of personnel required may be different for different maintenance tasks; and the exact composition of the recovery team or FRT may not be known. This adds significant complexity to the problem of selecting personnel. It is possible to write an input arc inscription to select either a multiset of fully-specified values of varying size, or a multiset of unspecified or partially specified values of fixed size. The authors are yet to identify a direct way to select a varying number of partially known data values on an input arc.

A personnel selection mechanism was implemented to overcome this problem. This mechanism is shown in Fig. 4, which depicts a fragment of the Assign Transport Resources page, showing only the Assign Recovery Assets and Assign FRT transitions. To overcome the first problem of selecing a varying number of tokens we instead select a fixed number of tokens (six in this model) using one variable per person. These variables, `p1` to `p6`, can be seen on the input arcs from Personnel to Assign Recovery Assets and Assign FRT. The timestamp on personnel tokens is the time at which they are due to go offline. The `Get_Soft_Limit` function on these two input arcs returns a time value specifying how long people are allowed to work in the usual course of events, and so this allows selection of people before they are due to go offline. The number six was chosen as a reasonable maximum size for recovery and forward repair teams, but if needed the personnel selection mechanism can be extended to cope with larger teams. The idea is that the six people selected are compared against what is required and those not required are returned to the Personnel place. This works provided no Recovery or Forward Repair Team requires more than six people, and that the Personnel place always has at least six personnel tokens in it (e.g. if we require only two people, there still needs to be at least six tokens in the Personnel place). To ensure this, we introduced the notion of `Dummy` peronnel, who have both trade and location equal to the string `Dummy`. They do no work and cannot be assigned to Recovery or Forward Repair teams, and hence always reside in the Personnel place. The Model Initialisation page inserts six such dummy personnel into the Personnel place.

To overcome the second problem (selecting a team of personnel of unknown composition) we have introduced a string constant, `Unspecified`. When we require a person but don't care what trade they are, we specify a requirement for an `Unspecified` trade, and match any trade (except `Dummy`) to `Unspecified`.

In [14] this mechanism was implemented using a guard and code segment totalling 50 lines of ML code for each of the Assign Recovery Assets and Assign FRT transitions. This mechanism has been substantially refined in our current model and is now implemented using a guard only. The sizable guard has been expressed more concisely using list manipulation functions within the `Check_Personnel_Assignment` function given in Listing 4. We describe our new implementation with respect to the Assign Recovery Assets transition. The Assign FRT transition works in a similar way.

The purpose of the `Determine_Required_Trades` function (shown in Listing 5) in the guard of Assign Recovery Assets is to return a list of six trades, comprising the actual trades required (these

could be Unspecified) for the Recovery team, obtained by invoking the Recovery_Personnel and Get_Min_Recovery_Population functions, with the balance made up of Dummy trades. For example, in a particular scenario, Recovery_Personnel may specify that a Recovery Team must contain one Recovery Mechanic, but Get_Min_Recovery_Population may specify a minimum size of 2 for a Recovery Team. The additional person could be any trade, so is specified by the Unspecified trade. The remaining four trades in the list of six trades would be Dummy. The function Check_Personnel_Assignment (Listing 4) then takes this list of six trades, all six selected people (variables p1 to p6 in a list), and the location of the request for maintenance, and checks that the people selected match the trades required and are at the correct location, or else are dummy personnel. Note that the matching of Unspecified with any trade except Dummy is shown on line 11.

Functions Remove_Dummy_Personnel and Remove_Non_Dummy_Personnel (not listed) have been added to the model to replace the code segments previously associated with Assign Recovery Assets and Assign FRT. As can be seen in Fig. 4 these two functions are used on output arcs to separate real personnel from dummy personnel as appropriate.

As was reported in [14] this model, which we denote Model 1, was found to perform reasonably well on small scenarios (10's of people) when cycling personnel online and offline, but performed very badly when assigning personnel to teams, due to the nature of the assignment and the selection mechanism described above. Standard ML profiling of simulation runs indicated that the two transitions in Fig. 4 were responsible for this poor performance. Specifically, we discovered that the 'less than' function automatically defined by CPN Tools for ordering the Personnel colour set was being executed of the order of 10^7 times, even for small scenarios, every time the enabling of these two transitions was checked. Because the Personnel place is connected to these two transitions (amongst others), every time the marking of the Personnel place changed, a check for the enabling of these two transitions was instigated. This resulted in a four to five second delay between successive steps of the simulation, where both transitions contributed equally to this delay. Hence we were able to determine that the time was being taken in the enabling check subsequent to the firing of a transition that affected the marking of the Personnel place, rather than firing the transition itself. For scenarios of increasing size, the delay quickly became prohibitive.

4.2 Personnel Modelled in a List

In an attempt to improve model performance, rather than modelling each person as a token, a list of personnel was considered. To do this, each person 'token' was augmented with a time value modelled as part of the colour set as shown in line 5 of Listing 6. This time value was used to store the timestamp that the person would have had as a single, separate token. The *personnel list* tokens were then given a timestamp corresponding to the smallest time value over all personnel in the list, to inform the model of the earliest time that at least one person becomes 'ready'.

Mechanisms were implemented in the models containing personnel in lists to retain the nondeterministic selection of personnel for Recovery and Forward Repair teams, and the nondeterministic selection of the next eligible person to move to/from an offline state. It was realised by the authors that for the purposes of this modelling exercise it was not necessary to capture this degree of nondeterminism for selection of the next person to move to/from an offline state. Hence two variations were also considered in [14]: one in which the head of the personnel list was (deterministically) selected as the person to

Listing 4. The Check_Personnel_Assignment function.

```
1  exception ListLengthException;
2
3  fun Check_Personnel_Assignment([],[],current_request_location) = true
4    | Check_Personnel_Assignment(required_trades,[],current_request_location) =
5    raise ListLengthException
6    | Check_Personnel_Assignment([],selected_personnel,current_request_location) =
7    raise ListLengthException
8    | Check_Personnel_Assignment((trade_required,ert)::rest_of_required,
9      (selected:Personnel)::rest_of_selected,current_request_location) =
10   (trade_required = #trade selected orelse
11     (trade_required = Unspecified andalso #trade selected <> Dummy))
12   andalso (if ((#trade selected) = Dummy)
13            then true else (#home_location selected)=current_request_location)
14   andalso (#working_status selected) = Ready
15   andalso Check_Personnel_Assignment(rest_of_required,rest_of_selected,current_request_location);
```

Listing 5. The `Determine_Required_Trades` function.

```
fun Determine_Required_Trades(personnel_to_assign: Job_List, minimumRequired) =
  let
    val personnel_needed = 6 - List.length(personnel_to_assign)
    val additional_personnel = minimumRequired - List.length(personnel_to_assign)
  in
    if personnel_needed = 0
    then personnel_to_assign
    else if  additional_personnel <= 0
    then  personnel_to_assign^^ms_to_list(personnel_needed`(Dummy,0))
    else  personnel_to_assign^^ms_to_list((additional_personnel`(Unspecified,0))
          ++ ((personnel_needed - additional_personnel)`(Dummy,0)))
  end;
```

Listing 6. The revised `Personnel` colour set for use when representing personnel in a list.

```
colset Person = record trade : Trade *
                       home_location : STRING *
                       working_status : Personnel_States *
                       last_came_online_time : INT timed;
colset Personnel = product Person * INT timed;
colset Personnel_List = list Personnel timed;
```

cycle offline/online; and the other in which all eligible personnel were cycled offline/online in one action. Further discussion of the merits of investigating these variations can be found in [14].

A list imposes an ordering on its elements. Two possibilities were considered in [14] when ordering the list of personnel: ordering the personnel values in the list firstly according to their time values and secondly according to the automatically defined 'less-than' function for the `Personnel` colour set; and ordering the personnel in the list according to time value only. Both of these orderings result in the earliest available person being at the head of the list, hence searching for available personnel in the list becomes at worst a linear operation (it is a constant operation if we only wish to find a single available person, regardless of trade, as may occur with technical inspections).

In [14], seven variations (Model 1, plus three models of personnel in lists with two variations on the ordering of elements within the personnel lists) were analysed. The most promising list model variation in terms of simulation performance was the model which cycled offline/online all eligible personnel in one action and ordered elements in the personnel lists by time value only. This model was denoted Model 4B in [14]. In the rest of this paper, the simulation performance of Model 1 and Model 4B are examined and compared.

5 Simulation Performance

When simulating Model 1 for a simple scenario comprising 90 personnel and 50 items of equipment distributed over 18 locations, the model proceeded relatively quickly (cycling personnel offline and online) up to the point where the first maintenance liability became due and a team of personnel was required. At this point, the simulation was manually terminated after waiting more than 24 hours of real time for personnel to be allocated to the first team.

As already mentioned, two of the key involvements of personnel are the cycling of personnel online and offline and the assignment of personnel to either a Recovery Team or a Forward Repair Team. In this section, the simulation performance of Model 1 and Model 4B with respect to these two activities is evaluated and discussed. Further, we compare the simulation performance of models that represent network topology explicitly in net structure (unfolded models) rather than encoding topology information within tokens (folded models). All experiments have been carried out with the Timed CPN Simulator of CPN Tools version 2.2.0 on a 2GHz Intel Core 2 Duo processor with 2Gb of memory.

5.1 Baseline Scenario

As a baseline, we consider a scenario comprising five highly simplified nodes, where each node comprises:

- two tradespeople at each node:
  - one Recovery Mechanic; and
  - one Vehicle Mechanic;

– two tasks that require teams to be formed at each node:
  • one Recovery Team, requiring a Recovery Mechanic and any other person; and
  • one Forward Repair Team, requiring a Vehicle Mechanic and any other person.

For the baseline scenario, this gives a total of 10 personnel and 10 tasks that require teams to be formed. (In the baseline scenario there will be a conflict between assigning personnel to the Recovery Team and to the FRT.) The number of teams that require personnel reflects the number of maintenance tasks in the system, which in turn reflects the number of pieces of equipment in the scenario. This is a more sophisticated baseline scenario than was considered in [14].

5.2 Specifying Tests of Model Performance

In an attempt to better understand the cause of the observed performance results, a series of tests was devised. The tests were designed to investigate the two different aspects of the operation of the model already mentioned (cycling personnel offline and online and the assignment of personnel to teams) as different aspects of the size of the scenario were scaled up. There are three dimensions to the size of scenarios that we investigate below: the number of personnel; the number of teams that simultaneously require assignment of personnel; and the number of nodes (maintenance workshops) in the scenario.

The process of assigning personnel to teams is affected by all three dimensions. The process of cycling personnel online and offline is affected only by the number of personnel and the number of nodes in the scenario, and is independent of the number of teams that require personnel. Hence, five different tests were considered. Only three tests were considered in [14]: cycling personnel online and offline as the number of personnel increases; and assigning personnel to teams as the number of teams increase and the number of personnel increase.

To accurately gauge the individual contribution of these two processes to the performance of the simulation, the following tests consider these two components in isolation[1] from the rest of the model.

Scaling the Number of Personnel To scale the personnel, we applied a *personnel multiplier* to the base number of tradespeople. By default, the personnel multiplier is one, giving 10 tradespeople in total (2 per node). A personnel multiplier of two gives 20 tradespeople in total (4 per node), and so on. The base number of teams and nodes remains unchanged, at 2 teams per node (10 in total) and 5 nodes respectively.

Scaling the Number of Tasks that Require Teams Scaling the number of tasks that require teams was done in a similar manner, using a *team multiplier*. Again, the default value of this multiplier is one, giving 10 tasks that require teams in total. The number of people and nodes remains unchanged, at 2 tradespeople per node (10 in total) and 5 nodes respectively.

Scaling the Number of Nodes Scaling the number of nodes was not considered in [14]. This required changes to the Model Initialisation page, hence there is no simple *node multiplier* per se. Rather, we refer directly to the number of nodes in the scenario. When scaling the number of nodes, each node is as described in the baseline scenario (2 tradespeople and 2 teams per node) unless otherwise stated.

Folded and Unfolded Network Topology To further understand the nature of the performance problems, we also examine the impact of having the network topology represented in data (a folded model) versus the network topology represented in net structure (an unfolded model). Again, this was not considered in [14], so the investigation of the impact of topology representation on simulator performance is an entirely new aspect considered in this paper.

The maintenance process captured by our model occurs within all maintenance nodes in the maintenance network. The idea of representing network topology by data was to avoid duplication of the net structure representing the process carried out at each node. Hence, a folded model brings with it the advantage of compactness and provides flexibility in terms of the scenario to be analysed, as the topology of the scenario is encoded in tokens. An unfolded model that represents topology explicitly

[1] These two processes were isolated by deleting all unrelated model pages and executable transitions.

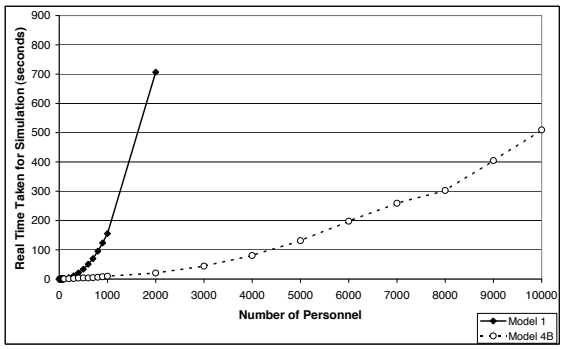

Fig. 5. Test 1: Performance of Personnel Cycling Offline and Online as No. of Personnel Increases.

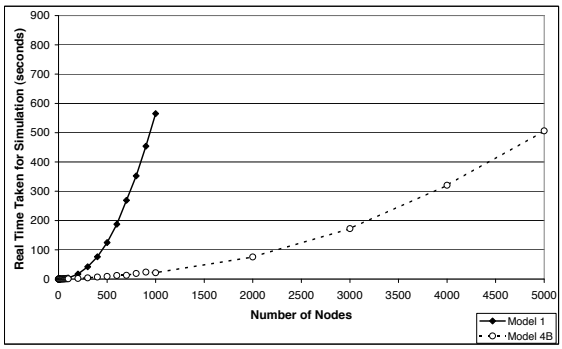

Fig. 6. Test 2: Performance of Personnel Cycling Offline and Online as No. of Nodes Increases.

in net structure requires modification of the net structure each time a new topology is analysed. However, unfolding the network topology has an advantage when specifying the maintenance network, as it provides direct spatial visualisation of the topology. This is helpful when the reader is trying to relate the model to the system (for validation) and for understanding materiel flows between different nodes. There is no similar benefit to be gained by unfolding the other data structures, such as equipment type, and doing so may result in a model that is less clear. Hence we decided to experiment with unfolding topology. From the analysis point of view, the motivation for producing an unfolded model is that the calculation of enabled binding elements becomes simpler. By unfolding the network topology, CPN Tools does not need to compute values for variables relating to different node locations, thus the number of computations is reduced.

To produce unfolded variants of the two sub-models considered in Section 5.3 we manually duplicated the page in question the number of times required (one page per node) and adjusted the Model Initialisation page accordingly. Hence, when scaling the number of nodes, we change the net structure. For this reason, we limit the number of nodes to range from 1 to 10 when analysing the unfolded models.

5.3 Performance of the Folded Models

For the five tests described above, we ran automatic simulations for 20 days of modelled time, whilst recording the real duration of the simulation. Results and discussion are presented below.

Test 1: Personnel Cycling Offline and Online when Increasing the Number of Personnel. Figure 5 shows a graph of the real time taken for each run as the personnel multiplier increases, for both models. Model 4B significantly outperforms Model 1 in this test (20.5 seconds vs 706.7 seconds at 1000 personnel, a factor of 34) confirming the result reported in [14]. The reason is most likely the ease with which an enabled binding element can be found for the two transitions that move personnel offline and online. There is only one list of personnel that are online, and one list of personnel that are offline, instead of a (potentially) large multiset of personnel from which to select.

However, we must be mindful of the fact that Model 4B's performance will deteriorate as the proportion of personnel eligible to move offline or online at the same model time decreases, e.g. as personnel availability become increasingly staggered due to performing maintenance work. There will be greater numbers of transition occurrences involving smaller numbers of personnel in each occurrence. The worst case corresponds to just the head of the personnel list being selected each time. Tests in [14] showed that this actually performs worse than Model 1. Hence, while Model 4B appears to be the clear winner in terms of performance when cycling personnel offline and online under the ideal circumstances of the test scenarios, it may deteriorate to perform worse than the original model.

Test 2: Personnel Cycling Offline and Online when Increasing the Number of Nodes. Figure 6 shows how the performance of the two models scales with the number of nodes when cycling personnel offline and online. The results look remarkably similar to those obtained when increasing the number of personnel (Fig. 5). At 1000 nodes, Model 1 took 565.1 seconds while Model 4B took 21.5 seconds - a factor of 26 improvement. This may be explained by considering what happens as the number of nodes

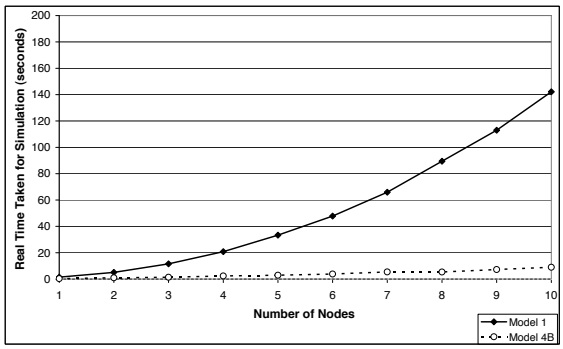

Fig. 7. Test 2*: Performance of Personnel Cycling Offline and Online as No. of Nodes Increases, with 100 Personnel per Node.

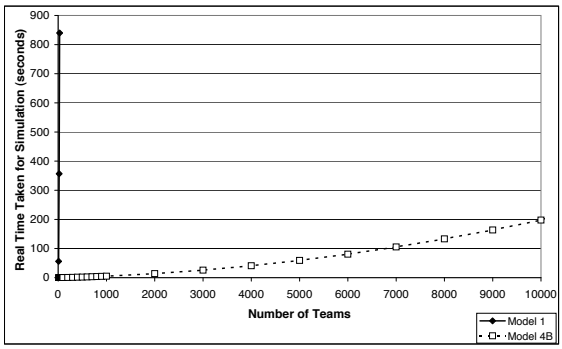

Fig. 8. Test 3: Performance of Assigning Personnel to Teams as No. of Teams Increases.

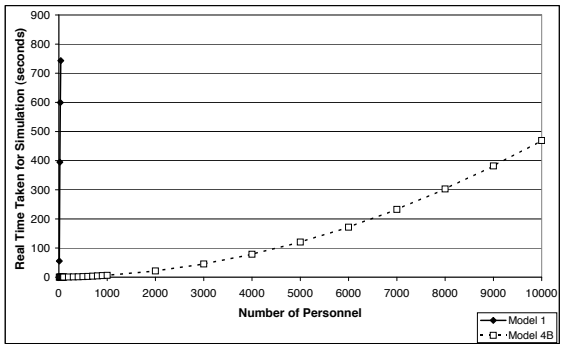

Fig. 9. Test 4: Performance of Assigning Personnel to Teams as No. of Personnel Increases.

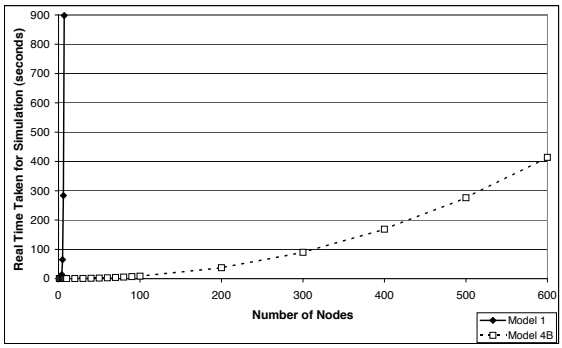

Fig. 10. Test 5: Performance of Assigning Personnel to Teams as No. of Nodes Increases.

increases. The nature of the model is such that although the personnel at each node may be conceptually geographically separate, they all appear on the same place in the model (the Personnel place) either as extra tokens (Model 1), or as extra elements in lists (Model 4B). Hence, the effect of increasing the number of nodes is the same as increasing the number of personnel.

This is not the case when considering the corresponding unfolded model, discussed later in Section 5.4. In order to allow direct comparison of results between the folded and unfolded models, we repeat Test 2 (as Test 2*) but only for the number of nodes from 1 to 10, where there are 100 personnel at each node instead of two (i.e. a personnel multiplier of 50). In this instance, because the range of nodes covered is small, we have increased the personnel multiplier so that the results obtained can be differentiated. These results are shown in Fig. 7, where we see that for 10 nodes, Model 4B (9 seconds) outperforms Model 1 (142.1 seconds) by a factor of 16.

Test 3: Assigning Personnel to Teams when Increasing the number of Teams. The first of three performance tests for assigning personnel to teams keeps the number of personnel fixed (personnel multiplier of 1) but increases the number of teams that simultaneously require personnel, by increasing the team multiplier. In order for the supply of available personnel at each node in the baseline scenario not to become exhausted before all teams are allocated (recall that there are only two tradespeople per node) personnel assigned to a team are immediately released back into the Personnel place, ready for assignment to another team. This is reasonable, given that we wish to analyse the process of assigning personnel to teams in isolation from the rest of the model, and hence we exclude the process of tradespeople performing maintenance before being released. The model was executed until all teams were assigned. Figure 8 shows a graph of the results obtained.

In this situation, Model 4B significantly outperforms Model 1. The three points shown for Model 1 are for 10, 20 and 30 teams. Assigning personnel to 30 teams took Model 1 839.9 seconds but Model 4B only 0.0625 seconds - a factor of 13400 improvement at 30 teams. (The factor of improvement is 6500

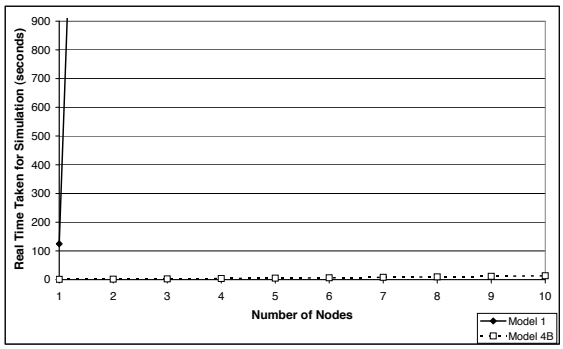

Fig. 11. Test 5*: Performance of Assigning Personnel to Teams as No. of Nodes Increases, with 200 Teams per Node.

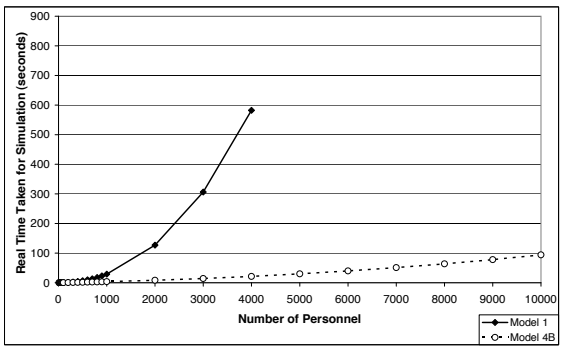

Fig. 12. Test 1^U: Performance of Personnel Cycling Offline and Online in an Unfolded Model as No. of Personnel Increases.

for 20 teams.) This may indicate how inefficient the mechanism for assigning a group of personnel of unknown composition and varying size implemented in Model 1 is.

Test 4: Assigning Personnel to Teams when Increasing the Number of Personnel. The second of the three tests for assigning personnel to teams keeps the number of teams requiring personnel fixed (at 10, 2 per node) but increases the number of personnel. A graph of this is shown in Fig. 9. Again, we see that Model 4B drastically outperforms Model 1. With 40 personnel to choose from, Model 1 took 743.2 seconds to assign personnel to all 10 teams, whereas Model 4B took only 0.101 seconds to do the same. This is a factor of 7350 improvement.

Test 5: Assigning Personnel to Teams when Increasing the Number of Nodes. Figure 10 shows the results obtained when scaling the number of nodes in the scenario. As in the previous two tests, the model of personnel using lists outperforms the baseline model of personnel as tokens. Model 1 took 898 seconds to assign personnel to teams in 7 nodes (2 teams per node), whereas Model 4B took only 0.0465 seconds - a factor of 19300 improvement at 7 nodes.

As was the case for Test 2, we also wish to repeat Test 5 for the purposes of comparison with the results from the equivalent unfolded model in Section 5.4. We again consider the number of nodes ranging from 1 to 10, but with 200 tasks requiring teams (requiring personnel) per node instead of 2 (a team multiplier of 100). We maintain the baseline of 2 tradespeople per node. The results of this test (Test 5*) are shown in Fig. 11. Only one point is shown on the graph for Model 1. This is for one node, where it took 125 seconds to assign personnel to all 200 teams. The next point on Model 1's curve is not shown, as for two nodes it took Model 1 5434 seconds (approx. 1.5 hours) to assign personnel to all 400 teams. Model 4B took only 1.67 seconds to assign personnel to all 400 teams, giving a factor of improvement of 3250 for two nodes. The increased number of teams has magnified the performance difference between Model 1 and Model 4B, contributing to the belief that Model 1's arbitrary personnel selection mechanism is highly sensitive to the number of tokens involved.

5.4 Performance of the Unfolded Models

Each test presented below is named according to the corresponding test in Section 5.3, but with a superscript 'U' to indicate that the test was carried out on the corresponding unfolded model. We have deliberately kept the X-axis scale of the graph the same as the corresponding graph in Section 5.3, to facilitate comparison. Note that in the case of Test 2^U the Y-axis scale has been reduced to 0-20 from 0-200 and in the case of Test 4^U the Y-axis scale has been reduced to 0-100 from 0-900 to avoid large areas of whitespace and to make the graphs easier to read.

Test 1^U: Personnel Cycling Offline and Online when Increasing the Number of Personnel. Figure 12 shows the results obtained when analysing the effect of increasing the number of personnel on the process of cycling personnel offline and online. Overall there was an improvement in computation time for both models for the unfolded case when compared to Fig. 5. For example, the unfolded Model

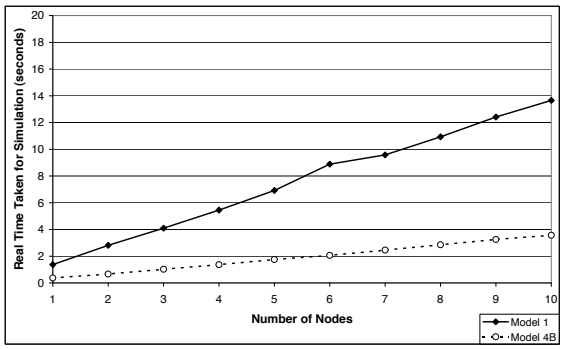

Fig. 13. Test 2^U: Performance of Personnel Cycling Offline and Online in an Unfolded Model as No. of Nodes Increases.

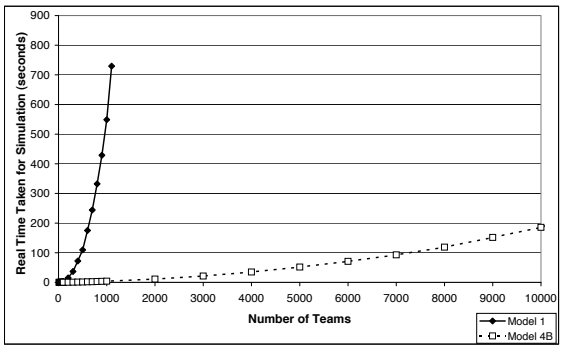

Fig. 14. Test 3^U: Performance of Assigning Personnel to Teams in an Unfolded Model as No. of Teams Increases.

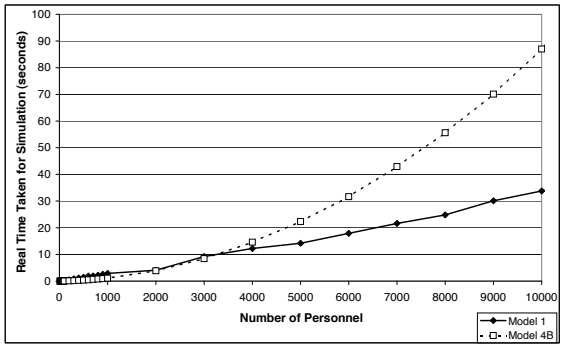

Fig. 15. Test 4^U: Performance of Assigning Personnel to Teams in an Unfolded Model as No. of Personnel Increases.

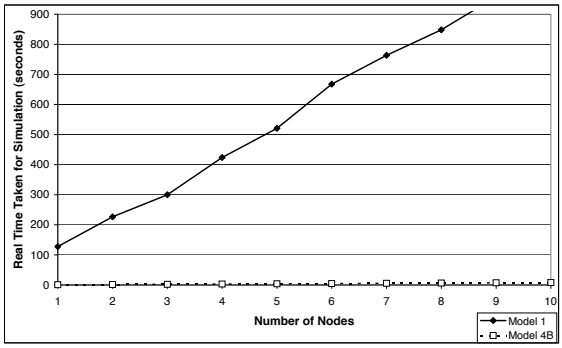

Fig. 16. Test 5^U: Performance of Assigning Personnel to Teams in an Unfolded Model as No. of Nodes Increases.

4B takes around 100 seconds to cycle 10,000 personnel offline and online for 20 model days, compared to 500 seconds for the folded Model 4B (Fig. 5) giving a factor of 5 improvement in performance due to unfolding. Model 1 has improved by approximately a factor of 6 at 2000 personnel. We suspect this is due to simpler calculations when determining enabled binding elements, but have yet to confirm this.

Test 2^U: Personnel Cycling Offline and Online when Increasing the Number of Nodes. This test reveals a major benefit when considering an unfolded model. Looking at Fig. 13, we see that the real time taken to cycle personnel offline and online scales linearly with the number of nodes. This is opposed to Fig. 7 in which real time taken appears to increase exponentially with the number of nodes. This linear growth is also evident in experiments that use a larger personnel multiplier (not shown in this paper). This linear behaviour makes sense intuitively, as each node is represented by a separate piece of non-interacting net structure. CPN Tools essentially repeats the same set of calculations for each separate node, hence one would expect the computational effort to be proportional to the number of nodes.

Test 3^U: Assigning Personnel to Teams when Increasing the Number of Teams. The results of this test are shown in Fig. 14. Unfolding the topology has not significantly affected model performance when modelling personnel in lists; indeed the curve for Model 4B shows only marginal improvement when compared to that in Fig. 8. However, whilst Model 4B still outperforms Model 1, there is a significant increase in the performance of Model 1. The gains in performance exhibited by Model 1 can be attributed to the fact that the number of tokens at any one place is reduced (distributed over multiple places), hence speeding up the calculations for enabling and firing of transitions. This effect is more pronounced for Model 1, where a reduction in the number of tokens to select from results in a more significant speed-up due to the inefficient mechanism for selection of personnel discussed in Section 4.1.

Test 4^U: Assigning Personnel to Teams when Increasing the Number of Personnel. This test (Fig. 15) reveals that, for the unfolded Model 1, the time taken to assign personnel to teams scales approximately linearly in the number of personnel. Figure 15 shows a significant improvement in performance when compared with Fig. 9. The folded Model 1 took over 700 seconds to assign personnel to teams when there were 40 personnel to choose from. The unfolded Model 1 took only 0.0785 seconds to do this. Indeed, there is a complete reversal of the performance trend, with Model 1 increasingly outperforming Model 4B as the number of personnel increases, despite Model 4B having improved in performance by a factor of 5 at 10,000 personnel.

Extending this experiment for Model 1 to 100,000 personnel (not shown) provides additional evidence of a linear relationship. This linear relationship may also be present for Model 1 in Fig. 9, however the increase in simulation time as the number of personnel increases is so severe that we cannot readily confirm this by experimentation. We conjecture that the mechanism in Model 1 for the selection of personnel is highly sensitive to the number of distinct personnel tokens from which to choose and the number of teams that require personnel, but is relatively independent of the multiplicity of the individual personnel tokens involved. Conversely, the list manipulation operations in Model 4B are affected by the length of the lists, i.e. the number of personnel under consideration. The length of the personnel lists increase regardless of whether the topology is folded or unfolded, the only difference being that when the topology is unfolded there are 5 lists of 'Ready' personnel (one per node) instead of a single list.

Test 5^U: Assigning Personnel to Teams when Increasing the Number of Nodes. In this test, the performance of both models scaled approximately linearly with the number of nodes. (Recall that each node has 200 teams that require personnel, a team multiplier of 100.) This is expected, since each new node is represented by its own piece of net structure. Figure 16 shows that Model 1 is also outperformed by Model 4B in this experiment. Despite this, Model 1 has shown dramatic improvement in performance. With one node, both the folded and unfolded models take approximately 125 seconds to assign personnel to all teams. With two nodes, the folded model took over 1.5 hours whereas the unfolded model took less than four minutes (225 seconds). Model 4B has improved, but not nearly as much. For 10 nodes the improvement is a factor of 1.68 (13.3 seconds folded vs. 7.94 seconds unfolded).

6 Discussion

We consider that model design for good model performance is difficult. The choice of modelling style, such as the choices for modelling personnel and topology discussed above, has a significant effect on the performance of the model. With respect to our original model (Model 1), we note that when there are complex interactions between personnel and tasks, the check for enabled binding elements results in a prohibitively large number of calculations. This has been greatly reduced for Model 4B, which performs well in most scenarios examined here. This is the case so long as personnel remain in step with respect to their scheduled offline and online cycle times. Otherwise, it may diminish performance rather than enhance it (compared to Model 1 - see [14] for more details).

When considering a folded network topology, modelling personnel as individual tokens results in highly inefficient simulations. A list-based representation appears to be computationally far superior without losing much of the desired behaviour of the token-based approach. Unfolding the network topology into net structure also improves the performance of the models, with Model 1 exhibiting the most significant improvement.

The above tests purposely cover extreme values of personnel, teams and nodes to elicit performance trends. For realistic scenarios we can see that Model 4B provides an acceptable level of performance for the case of cycling personnel and assignment of personnel to teams in isolation. However, this may not be the case when considering the model as a whole, as it is difficult to directly infer the performance of the complete model from these results. We are currently investigating the performance of both approaches to modelling personnel in the complete model.

Unfolding other aspects of the model may provide significant performance gains, but it is a significant undertaking and there comes a point at which the unfolded model becomes unmanageable. We must also consider the loss in flexibility that moving to an unfolded model would impose. For example, it would be useful to specify the tool facilities that could be developed to produce a new model for each distinct topology that needs to be simulated. Producing (partially) unfolded versions of the complete model for one or two carefully chosen scenarios may help to clarify the advantages and disadvantages.

We would also like to investigate a way of modelling personnel changing state from online to offline, and vice versa, that does not involve dedicated transition occurrences. Because the cycling of personnel is predetermined (in the absence of maintenance tasks) it follows that by knowing the most recent maintenance activity of all personnel, all future availabilities (until the next maintenance activity starts) can be derived. Therefore, explicit modelling of the transitions of personnel online and offline may not be necessary.

7 Conclusions

This paper has explored different modelling approaches to improve the performance of CPN Tools when simulating an industrial-scale CPN model that captures the Australian Army's maintenance process. Our aim was to improve simulation performance to a level that is suitable for timely evaluation of different maintenance scenarios.

In its usual operating environment, the maintenance system involves hundreds of personnel and thousands of pieces of equipment distributed over tens of locations. Under these conditions, simulations of our original model did not allow any results to be obtained within an hour. This led us to profile the model to determine where the bottlenecks in the simulation occurred. We found that the internal CPN Tools' function, 'less than', for the personnel colour set was being executed millions of times during the checking of the enabling of two transitions, both associated with assigning personnel to teams. Given this, we decided to explore different data structures for personnel. We have made some dramatic performance gains over our original model by using lists of personnel in the maintenance process, rather than the more natural use of multisets. Two components of the model (cycling personnel online and offline, and assigning personnel to teams) are now nearing the levels of performance required to make the model fast enough to be useful in supporting simulated military operations.

We have also considered models in which the network topology of the maintenance system is not encoded in tokens but rather is represented explicitly in the net structure (i.e. a partial unfolding of the CPN). This approach has also shown promise, opening up many possibilities to further enhance the performance of the model.

It is likely that the performance issues discussed in this paper will be present in many industrial-scale CPN models. We therefore believe that a more fundamental understanding of the relationship between the use of various modelling constructs and their impact on analysis and simulation performance in CPN Tools will be beneficial to the user community, particularly those concerned with industrial systems. We hope that the results in this paper will provide a starting point for the development of a set of guidelines for modelling complex systems that are more readily simulated by CPN Tools.

Acknowledgments

The authors would like to acknowledge their colleague, Mr. Christopher Moon, for his involvement in the early development of this model, and Dr. Nimrod Lilith, for support in developing some aspects of the model and for participation in many active discussions. The authors would also like to acknowledge Dr. Lisa Wells and Prof. Lars Kristensen for productive discussions and information regarding profiling. Finally, the authors would like to acknowledge the constructive feedback received from the anonymous reviewers of [14].

References

1. W.M.P van der Aalst (guest editor). *Transactions on Petri Nets and Other Models of Concurrency II, Special Issue on Concurrency in Process-Aware Systems*. Lecture Notes in Computer Science. Springer, 2009.
2. J. Ashayeri, A. Teelen, and W. Selen. Production and maintenance planning model for the process industry. *International Journal of Production Research*, 34(12):3311–3326, 1996.
3. J. Billington, M. Diaz, and G. Rozenberg, editors. *Application of Petri Nets to Communication Networks*, volume 1605 of *Lecture Notes in Computer Science*. Springer, 1999.
4. L. Briskin and W. S. Demmy. An overview of the network repair level analysis model [military systems]. In *Proceedings of the IEEE 1988 National Aerospace and Electronics Conference (NAECON 1988)*, volume 4, pages 1414–1420, 1988.
5. C. R. Cassady, W. P. Murdock, and E. A. Pohl. Selective maintenance for support equipment involving multiple maintenance actions. *European Journal of Operational Research*, 129(2):252–258, 2001.

6. T. N. Cook and R. C. DiNicola. Modeling Combat Maintenance Operations. In *Proceedings of the Reliability and Maintainability Symposium*, pages 390–396, 1984.
7. CPN Tools. `http://www.cs.au.dk/CPNTools`.
8. DSTO. *Coloured Petri Net Modelling of Defence Logistics*. Australian Government, Department of Defence, Contract No. 4500498737, 1 February 2006.
9. W. W. Fisher. Markov process modelling of a maintenance system with spares, repair, cannibalization and manpower constraints. *Mathematical and Computer Modelling*, 13(7):119–125, 1990.
10. O. Fouathia, J.-C. Maun, P.-E. Labeau, and D. Wiot. Stochastic approach using Petri nets for maintenance optimization in Belgian power systems. In *Proceedings of 8th International Conference on Probabilistic Methods Applied to Power Systems*, pages 168–173, 2004.
11. G. E. Gallasch and J. Billington. Army Maintenance System Analysis Tool Enhancement: CPN Model Documentation. Technical Report CSEC-39, Computer Systems Engineering Centre Report Series, University of South Australia, September 2009.
12. G. E. Gallasch, N. Lilith, and J. Billington. Extending a Coloured Petri Net Model of a Defence Logistics Network. Technical Report CSEC-29, Computer Systems Engineering Centre Report Series, University of South Australia, September 2007.
13. G. E. Gallasch, N. Lilith, J. Billington, L. Zhang, A. Bender, and B. Francis. Modelling Defence Logistics Networks. *International Journal on Software Tools for Technology Transfer, special section on CPN'06*, 10(1):75–93, 2008.
14. G. E. Gallasch, C. Moon, B. Francis, and J. Billington. Modelling personnel within a defence logistics maintenance process. In *Proceedings of 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2008)*, March 2008. (10 pages).
15. T. Gossard, N. Brown, S. Powers, and D. Crippen. Scalable Integration Model for Objective Resource Capability Evaluations (SIM-FORCE). In *Winter Simulation Conference Proceedings*, volume 2, pages 1316–1323, 1999.
16. A. Jacopino, F. Groen, and A. Mosleh. Modelling imperfect inspection and maintenance in defence aviation through Bayesian analysis of the KIJIMA type I general renewal process (GRP). In *Proceedings of Reliability and Maintainability Symposium (RAMS'06)*, pages 470–475, 2006.
17. K. Jensen. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Volumes 1 to 3, Basic Concepts, Analysis Methods, and Practical Use*. Monographs in Theoretical Computer Science. Springer, 2nd edition, 1997.
18. K. Jensen, L. M. Kristensen, and L. Wells. Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer*, 9(3-4):213–254, June 2007.
19. K. Jensen and L.M. Kristensen. *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*. Springer, 2009.
20. W. D. Kelton, R. P. Sadowski, and D. T. Sturrock. *Simulation with Arena*. McGraw-Hill Higher Education, 3rd edition, 2004.
21. L. M. Kristensen, P. Mechlenborg, L. Zhang, B. Mitchell, and G. E. Gallasch. Model-based Development of a Course of Action Scheduling Tool. *International Journal on Software Tools for Technology Transfer, special section on CPN'06*, 10(1):5–14, 2008.
22. O. Maimon and M. Last. Information-efficient design of an automatic aircraft maintenance supervisor. *Computers & Operations Research*, 20(4):421–434, 1993.
23. V. Mattila and K. Virtanen. A simulation-based optimization model to schedule periodic maintenance of a fleet of aircraft. In *Proceedings of European Simulation and Modelling Conference 2005 (ESM'2005)*, pages 479–483, 2005.
24. B. Mitchell, L. M. Kristensen, and L. Zhang. Formal Specification and State Space Analysis of an Operational Planning Process. In *Fifth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pages 1–17. Department of Computer Science, University of Aarhus, 2004. Available via `http://www.daimi.au.dk/CPnets/workshop04/cpn/papers/`.
25. S. R. Parker. Combat Vehicle Reliability Assessment Simulation Model (CVRASM). In *Winter Simulation Conference Proceedings*, pages 491–498, 1991.
26. T. Raivio, E. Kuumola, V. A. Mattila, K. Virtanen, and R. P. Hamalainen. A simulation model for military aircraft maintenance and availability. In *Proceedings of Modelling and Simulation 2001, 15th European Simulation Multiconference (ESM'2001)*, pages 190–194, 2001.
27. M. Ramadass, J. Rosenberger, B. Huff, S. Gonterman, and R. N. Subramanian. Simulation Modelling for a Bus Maintenance Facility. In *Winter Simulation Conference Proceedings*, pages 1222–1225, 2004.
28. J.D. Ullman. *Elements of ML Programming*. Prentice-Hall, 2nd edition, 1998.
29. L. Zhang, L. M. Kristensen, B. Mitchell, G. E. Gallasch, P. Mechlenborg, and C. Janczura. COAST - An Operational Planning Tool for Course of Action Development and Analysis. In *Proceedings of the 9th International Command and Control Research and Technology Symposium (ICCRTS), Copenhagen, Denmark.*, 2004.