Resource-Oriented Timed Workflow Nets and Simulation Tool Design

Yuansi Hu and Jiacun Wang, *Senior Member, IEEE*
Monmouth University, New Jersey, USA
Guanjun Liu
Tongji University, Shanghai, China

Abstract—This paper introduces resource-oriented timed workflow nets (ROTWNs), a type of Petri nets extended with resources and timing specifications. ROTWNs are designed to facilitate the analysis of the resource requirements in business workflow, as well as the timespan of the process. The analysis of ROTWN models is presented. Due to the nature of concurrency of business processes, state explosion is a common issue that makes it hard to find analytical solutions to ROTWN models, and thus it is necessary to automate the analysis through simulation. The design of such a tool is explained in this paper as well. A toy example of 4-task workflow is also used to illustrate the concept and analysis of ROTWNs.

***Keywords*—Workflow, timed Petri nets, resource requirements, timeliness analysis, simulation.**

I. Introduction

Workflow technology has been wildly used for the automation of business processes [1][2]. A workflow model with resource usage defined can be used to track the availability of various resources and analyze resource requirements for the workflow execution [18]. Furthermore, a workflow that is augmented with task execution time can provide a base for the timeliness analysis of emergency response processes.

Being rigor and unambiguous, formal methods have been employed in software and hardware system specifications [16][20]. Among them, Petri nets are well-known for intuitive graphic representation and strong power in modeling various discrete event dynamic systems [6][14][15][17]. For the same reason, Petri nets have also found wide applications in workflow modeling and analysis [4][5][8]. Of course, simulation is always a powerful tool for system evaluation. For example, Brachner presented a simulation model to evaluate an emergency response system for offshore helicopter ditches in [3]. In [8], Filippouplitis presented a simulation system that models the evacuation of a multi-story building using wireless sensor networks. A simulation engine for stochastic timed Petri nets and application to emergency healthcare systems is presented in [24].

Petri nets are also a powerful tool for the modeling and analysis of resources-constrained systems. Resource allocation and deadlock avoidance are two popular research topics. For example, resource-constrained workflow nets are introduced and discussed in [9][10][11][13]. In these works, the authors studied extensions of workflow nets in which processes can share global resources. An important assumption in resource-constrained workflow nets is all resources are durable, they can be used and released, but never be created or destroyed.

In [19][21], a workflow intuitive yet formal approach (WIFA) was developed for emergency response workflow modeling and resource analysis. In [18], the concept of resource-oriented workflow nets (ROWN) was introduced. It is a Petri net-based model. Resource requirements for general workflow can be done through reachability analysis. For a class of well-structured workflows, an efficient resource analysis algorithm was also developed. ROWNs have been applied to emergency healthcare staffing and resource requirement analysis [22][23][24].

The resource-oriented timed workflow nets (ROTWN), a type of Petri nets, are introduced in this paper to facilitate the analysis of the resource requirements in an emergency response, as well as the timespan of the process. Algorithms that work for workflows with most common structural constructs, such as sequence, concurrency and competition are developed. This paper focuses on the development of a simulation tool, which supports the analysis of large-scale emergency response workflows.

The paper is organized as follows: Section II introduces the ROTWN model and its state transition rules. Section III discusses the analysis of ROTWN model. The development of simulation tool is presented in Section IV. An example on the ROTWN is given in section V. Finally, Section VI presents conclusions and ideas for the continuation of the work.

II. Resource-Oriented Timed Workflow Nets

A workflow is composed of *tasks* that execute in a specific order. A task is an activity or event. A workflow net is a special type of Petri net [2]. A resource-oriented timed workflow net (ROTWN) is a workflow net in which each transition is associated with

- a vector to reflect resource changes when it fires, and
- a predefined exponentially distributed firing time or a constant firing probability.

The benefit of assuming exponential transition firing times is in its simplicity due to the memoryless property of the exponential distribution [12]. However, this restriction can be relaxed in the design of simulation tools. An exponential distribution can be uniquely specified by its rate parameter. The formal definition of ROTWNs is as follows:

An ROTWN is an 8-tuple: $ROTWN = (P, T, I, O, \Lambda, R, M_0, S_0)$, in which

- (P, T, I, O, M_0) is a workflow net.
- $T = T_{tim} \cup T_{ins}$. Transitions in T_{tim} are *timed* transitions that have firing delays and those in T_{ins} are *immediate* transitions that can fire instantly. $T_{tim} \cap T_{ins} = \varnothing$.
- $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ is a vector of firing parameters. If $t_k \in T_{tim}$, λ_k is the *firing rate* of t_k. Otherwise, $t_k \in T_{ins}$, λ_k is the *assigned firing probability* of t_k.
- $R = (R_1, R_2, \ldots, R_n)$, where R_k is a vector that defines the resource change when the transition t_k fires. $R_k \in \Re^h$, where $\Re$ is the set of real numbers and h the number of types of resources.
- $A_0 \in (\Re^+)^h$ represents the initially available resources.

In order to catch the resource changes when a task starts and finishes, we use two sequential transitions to model a task, one modeling the beginning of the task execution and the other the end of the task execution, as shown in Fig. 1. Transition B is associated with a positive vector R_B and transition E is associated with a negative vector R_E. A token in p_1 indicates that the task is idle; a token in p_2 means the task in execution.

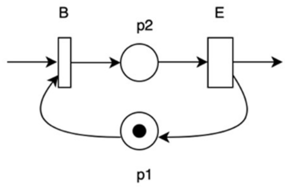


Fig. 1. Petri model of a task.

A. States

A *state* of an ROTWN is composed of two elements:

- the marking, and
- the available resources.

It is denoted by $S = (M, A)$. Different states may have the same marking.

A transition t_k is *enabled* at the state $S = (M, A)$ if and only if

$$M \geq I(t_k), \tag{1}$$

$$A \geq R_k \tag{2}$$

Inequality (1) is the same as it is with regular Petri nets. Inequality (2) is to ensure there are enough resources to support the event.

Let the set of all enabled transitions at the state S be denoted by $E(S)$, the selection of a firing transition from $E(S)$ proceeds according to the following three cases:

Case 1: $E(S) \subseteq T_{tim}$. A sample of random firing time is taken from each individual exponential distribution. The smallest sample determines the transition to fire and the actual firing delay.

Case 2: $E(S) \subseteq T_{ins}$. A default assumption is that the firing probabilities of all these enabled transitions constitute a probability distribution. In other words, they add up to 1. Therefore, the firing transition can be simply selected by generating a random number according to the distribution.

Assume the underlying Petri nets are free-choice net only. In Case 2, if at a state, multiple decision sets are enabled, then the total firing probabilities will be added up to more than 1. The firing probability of each individual t_i must be *adjusted* based on the following formula:

$$\lambda_i' = \frac{\lambda_i}{\sum_{t_j \in E(S) \setminus T_{tim}} \lambda_j} \tag{3}$$

If one or more transitions in a decision set are not enabled due to lack of sufficient resources and the sum of the firing probabilities of the enabled transitions is less than 1, we need to recalculate the adjusted firing probabilities as follows:

$$\lambda_i' = \frac{\lambda_i}{\sum_{t_j \in D \setminus D'} \lambda_j} \tag{4}$$

where D is the set of immediate transitions in the decision set, and D' is its subset in which transitions are not enabled due to lack of resources.

Case 3: $E(S) \cap T_{tim} \neq \varnothing$ and $E(S) \cap T_{ins} \neq \varnothing$, i.e., the enabled transitions include both timed ones and immediate ones. In this case, we follow Case 2 to select the transition to fire from the immediate transitions.

B. State Transitions

Assume after the firing of transition t_k at the old state S, a new state, $S' = (M', A')$, is reached. M' and A' are calculated as follows:

$$M' = M + O(t_k) - I(t_k), \tag{5}$$

$$A' = A - R(t_k). \tag{6}$$

Note that when we model a task, if t_k represents that a task that has been executed, $R(t_k)$ is a negative vector, and thus $A - R(t_k)$ will add (return) resources to A'.

Based on this state transition rule, reachability analysis can be performed, which explores all possible states and execution paths, and calculates the resource requirements and timespan of the workflow execution. The process is similar to the reachability analysis of regular Petri nets.

III. ROTWN Analysis

Reachability analysis is the fundamental way exploring the properties of a Petri net model. This is also the case for ROTWNs. Reachability analysis on an ROTWN can reveal all reachable states, feasible execution paths, and state-by-state resource changes along all execution paths. The ROTWN reachability analysis is conducted by generating the reachability tree.

A. Resource Requirements

Based on an ROTWN model, one can analyze the modeled system resource requirement in several ways:

(1) Find the requirement of resources in executing a particular workflow path. This can be done by tracking the

availability of each type of resources and recording its lowest level.

(2) Given a set of resources, find all feasible paths to execute the workflow. This can be easily done through reachability analysis. We construct the reachability tree of an ROTWN first, and then we examine each leaf node (state) of the tree and find all leaf nodes that indicate a completion of the workflow. All paths from the root (initial state) to these leaf nodes are feasible paths.

B. Path Probability

A workflow can have many feasible execution paths and it is important to find out resource reequipments along each single execution path. Each path is executed with different probability. Consider a feasible path

$$\sigma = t_1 t_2 \ldots t_f$$

$$\Gamma(\sigma) = S_0 S_1 S_2 \ldots S_f$$

Transition t_k, $1 \le k \le f$, is enabled in $S_{k\text{-}1}$. If t_k is an immediate transition, its actual firing probability is a known parameter. If t_k is a timed transition, then, because all enabled transitions in $E(S_{k\text{-}1})$ must be timed transitions and their firing times are exponentially distributed, the firing probability of t_k is given by

$$\frac{\lambda_k}{\sum_{t \in E(S_{k-1})} \lambda}$$

The path probability is product of each transition's firing probability.

C. Timeliness Analysis

Given a state of an ROTWN mode, if the firing transition is an immediate transition, the state changes instantly. If it is a timed transition, however, the state will have a tangible exponentially distributed sojourn time. The most important property of exponential distribution is that it is memoryless, which is formally defined as

$$\Pr(\tau > r + s) = \Pr(\tau > s) \quad (7)$$

where τ is a random variable, $r > 0$ and $s > 0$. With this property, if a timed transition was enabled in the previous state but didn't fire, and in the current state it is still enabled, its firing time distribution in the current state is exactly the same as if it just became enabled. Therefore, for all enabled timed transition, no matter when they became enabled, we can simply treat them as they are freshly enabled.

Without loss of generality, assume along an execution path σ, there are l timed transitions and their firing times are denoted by $\tau_1, \tau_2, \ldots \tau_l$. The sum of these firing times, which is also the path execution time of the path, is denoted by Q, i.e.

$$Q = \sum_{k=1}^{l} \tau_k \quad (8)$$

The mean of Q is

$$\boldsymbol{E}(Q) = \sum_{k=1}^{l} \lambda_k^{-1} \quad (9)$$

Assume all tasks' execution times are uncorrelated. Then all the covariances are 0, and we have

$$\boldsymbol{Var}(Q) = \sum_{k=1}^{l} \boldsymbol{Var}(\tau_k) = \sum_{k=1}^{l} \lambda_k^{-2} \quad (10)$$

IV. Simulation Tool

Because of the existence of concurrency in typical Petri net models, state space explosion is unavoidable, which makes it hard to get analytic results through thorough reachability analysis. Therefore, developing a software tool to support the automation of the resource and timeliness analysis of ROTWN models is necessary. We developed such a tool in Java.

The workflow of the simulation tool is shown in Fig. 2. A JSON file that specifies all ROTWN elements is first read into the simulation tool and the ROTWN model is then stored in the internal data structure of the tool. Then the simulation engine will run and compute the resource requirements and timing performance. The simulation results will be displayed and stored in a CSV file.

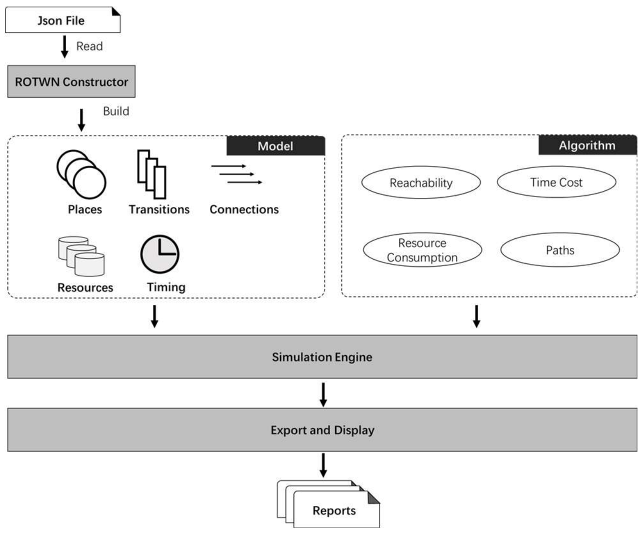


Fig. 2. Overview of the ROTWN simulation tool.

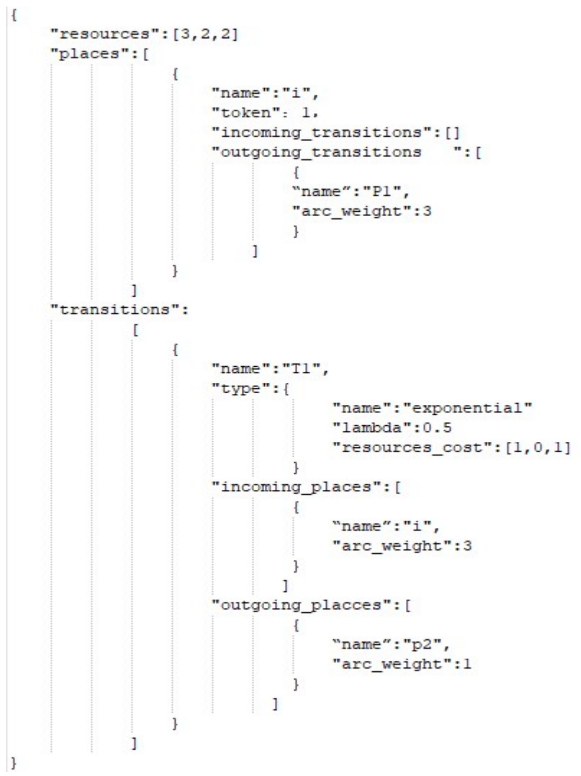
A. ROTWN Definition in JSON

An example JSON file that defines an ROTWN model is shown in Fig. 3. There are three types of objects: resources, places and transitions. The resources object specifies the initial count of all resources a system has; The places object defines all places in the model. For each place, an name, a number of tokens, an array of incoming transitions and an array of outgoing transitions are stated. The transitions object specifies all transitions. For each transition, a name, a type, an array of incoming places and an array of outgoing places are stated. Specifically, for the type object of the transition, we support three types: exponential, uniform and probability. For the exponential type, one needs to specify the name to be exponential, the lambda parameter of the exponential distribution and the cost of resources. For uniform distribution, one needs to specify the name to be uniform, the lower bound, the upper bound of the uniform distribution and the cost of resources. For probability, one needs to specify the name to be probability, the value of probability and the cost of resources.

B. Algorithm Design

In the high level, the simulation proceeds with reachability analysis. However, instead of computing all possible execution paths, the simulation only considers one path a time. When an execution path is built up, the resource requirements and timing performance of the workflow under investigation are evaluated. Due to the space limitation, here we only present the overall control flow and the algorithm to select firing transitions.

It defines how the firing sequence is determined and whether the token will end up in the exit place. It contains a fireable transition algorithm which calculates a list of transitions that are able to run with minimum time elapses. The time cost algorithm defines how time cost is calculated. The resource consumption algorithm defines how the maximum resources cost are calculated. The paths algorithm defines how the possible paths, and their odds are calculated.



```
{
    "resources":[3,2,2]
    "places":[
            {
                "name":"i",
                "token": 1,
                "incoming_transitions":[]
                "outgoing_transitions    ":[
                        {
                        "name":"P1",
                        "arc_weight":3
                        }
                    ]
            }
        ]
    "transitions":
        [
            {
                "name":"T1",
                "type":{
                        "name":"exponential"
                        "lambda":0.5
                        "resources_cost":[1,0,1]
                        }
                "incoming_places":[
                        {
                        "name":"i",
                        "arc_weight":3
                        }
                      ]
                "outgoing_placces":[
                        {
                        "name":"p2",
                        "arc_weight":1
                        }
                    ]
            }
        ]
}
```

Fig. 3. JSON specification of ROTWN elements.

Algorithm – Simulation control flow

```
Input: An ROTWN Model
Output:
  A feasible execution path with resource requirements
  and timing performance
Initialization:
  simTime = 0   //simulation time
  timeIncrement = 0  //simulation step length
  availResources = initial resources    //available resources
  fireableTransition = null
  execPath = null   //execution path
Main Flow:
  while there are fireable transitions
      select nextFiringTrans;
      timeIncrement = the firing time of nextFiringTrans;
      simTime+ = timeIncrement;
      fire nextFiringTrans;
      add nextFiringTrans to execPath;
      update resources;
      add resources to resSeq;
      calculate fireable transitions in the new state;
   compute resource requirements based on resSeq;
   record simTime
```

Algorithm – Select next transition to fire

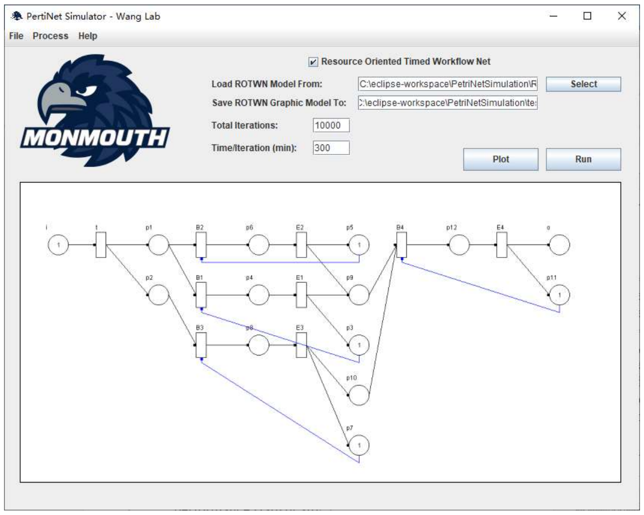
```
Input: An ROTWN and a state S.
Output: Next firing transition.
identify all enabled decision sets D.
for each D,
   if sum of immediate trans probabilities < 1
      adjust each trans' probability according to Eq. (4);
   If sum of immediate trans probabilities > 1
      adjust each trans' probability according to Eq. (5);
identify all enabled timed transitions;
select the firing transition according the probability
distribution among all enabled transitions.
```

The tool allows users to input the number of simulation times in order to get the resource requirements and timing performance statistically.

C. Graphical User Interface

The graphical user interface of the simulation tool is shown in Fig. 4. There are three sections for the simulation tool – model settings, topology display and result display.

The model settings section consists of input box for configuration file path, configuration file selector, input for the schematics output path, input box for total iterations to be executed, time spent for each iteration, radio button for ROWTN, and result for standard petri net simulation. Either the input box or the selector can be used to specify the path to the configuration file that defines the topology of the model. The schematics output path is used to save the schematics to when the plot button is pressed. Total iterations are used with ROTWN radio button – when a ROTWN simulation is executed, it determines how many iterations the analysis will be performed. If ROTWN is not selected, the time spent will specify how long the Petri Net simulation can be executed and the result of average time cost will be displayed in the result label.



PertiNet Simulator - Wang Lab

[✓] Resource Oriented Timed Workflow Net

Load ROTWN Model From: C:\eclipse-workspace\PetriNetSimulation\R

Save ROTWN Graphic Model To: :\eclipse-workspace\PetriNetSimulation\te:

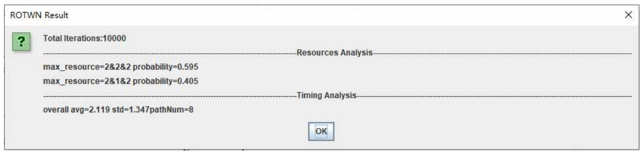
Total Iterations: 10000

Time/Iteration (min): 300

Fig. 4. GUI of ROTWN simulation tool.

D. Result Display and Export

The result display section is a popup window when the ROTWN is selected. An example of the report is shown in Fig. 4. In the top of the report, it displays how many iterations have been executed and, in this case, it is 10. In the second section, it displays resource consumption for each and every different path executed. For example, in this case, there are 60% of the chance that the max resource consumption for resources x, y and z are 2, 2, 2 and there are 40% of the chance that the max resource consumption for resources x, y and z are 2, 1, 2. In the last section, it summarizes all the average time cost, the standard deviation and all possible number of paths.



```
ROTWN Result

Total Iterations:10000
----------------------------------------Resources Analysis----------------------------------------
max_resource=2&2&2 probability=0.595
max_resource=2&1&2 probability=0.405
----------------------------------------Timing Analysis----------------------------------------
overall avg=2.119 std=1.347pathNum=8
```

Fig. 5. Example simulation results.

In addition to the information that is displayed in the pop-up window, which is illustrated in Fig. 5, the software will also generate a detailed report on the simulation executed and export it to the local directory where the software resides in.

V. AN EXAMPLE

Assume a work flow is composed of four tasks, namely A, B, C and D. When the workflow starts, either A or B, and C will start to execute immediately. When A or B is finished, and C is finished as well, D will be executed. When D is finished, the workflow is completed. There are three types of resources involved in the workflow execution, namely *x*, *y* and *z*. Resource requirements are listed in Table 2. Initial available resources are {2'*x*, 3'*y*, 2'*z*}. When the workflow starts, the probability of A getting executed is 0.6 and that of B getting executed is 0.4. All the four tasks take exponential time to finished. The exponential distribution parameters of A, B, C and D are 0.2, 0.1, 0.25, and 0.05, respectively.

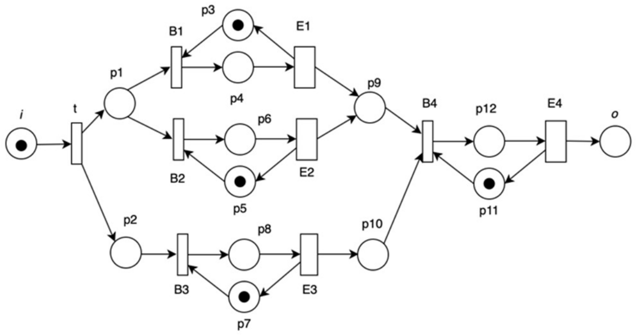


Fig. 6. ROTWN model of a workflow with four tasks.

Table 2: Resources for tasks in the example workflow.

Task	Resource required for execution	Resource released after execution
A	1'*x*, 2'*y*	1'*x*, 1'*y*
B	1'*x*, 1'*y*	1'*x*
C	1'*x*, 2'*z*	2'*z*
D	1'*z*	1'*z*

The ROTWN model of this workflow is shown in Fig. 6. Places *i* and *o* are source and sink places, respectively. The transition *t* marks the start of the workflow. Transitions B1, B2, B3 and B4 represent that tasks A, B, C and D start execution, respectively, while E1, E2, E3 and E4 represent that A, B, C and D end execution, respectively. Places p_3, p_5, p_7 and p_{11}, when containing a token, each represents the corresponding task is idle. According to Table 2, resource change vector of all the previously mentioned transitions are as follows:

t: (0, 0, 0)	B1: (1, 2, 0)	E1: (-1, -1, 0)
B2: (1, 1, 0)	E2: (-1, 0, 0)	B3: (1, 0, 2)
E3: (0, 0, -2)	B4: (0, 0, 1)	E4: (0, 0, -1)

There are five immediate transitions. The firing probabilities of B1 and B2 are 0.6 and 04, respectively. For all the other three, the probability is 1. The firing rates of E1, E2, E3 and E4 are 1, 2, 1 and 3, respectively.

Initially available resources are A_0 = (2, 3, 1). Initial marking M_0 is that places *i*, p_3, p_5, p_7 and p_{10}, each has a token; none of the other places has tokens. Since the workflow net is a safe Petri net, we use a sequence of names of marked places to specify a marking. For example, $M_0 = ip_3p_5p_7p_{10}$.

For the ROTWN model in Fig. 6, we ran the simulation for 10,000 times and the paths, path probabilities and response time parameters are listed in Table 1:

Table 1. Path probability and timing parameters.

No	Firing Sequence	Path Prob.	Mean	STD
1	t B3 B1 E1 E3 B4 E4	0.1698	21.390	13.071
2	t B3 B2 E3 E2 B4 E4	0.0345	10.541	6.748
3	t B1 B3 E3 E1 B4 E4	0.1042	16.372	9.928
4	t B3 B1 E3 E1 B4 E4	0.1017	17.052	9.952
5	t B2 B3 E3 E2 B4 E4	0.0322	10.580	6.683
6	t B2 B3 E2 E3 B4 E4	0.1671	17.983	10.971
7	t B1 B3 E1 E3 B4 E4	0.1941	22.011	13.002
8	t B3 B2 E2 E3 B4 E4	0.1694	18.565	12.194

Among the eight paths, paths 1, 3, 4 and 7 require the same quantity of resources, which is (2 2 2), while the requirement of the other four paths is (2 1 2).

If we change the initial available resources to {2'x, 1'y, 2'z}, i.e. $A_0 = (2, 1, 2)$, the simulation will only report four paths, which are the same as the paths 2, 5, 6 and 8 listed in Table 3. The other four paths are halted in the middle of execution due to lack of sufficient resources.

VI. Concluding Remarks

This paper presented resource-oriented timed workflow nets (ROTWNs), a powerful formal model that allows users to analyze the resource requirements of emergency response, tracks the resource level, and estimate the timespan of the response process. This new model allows the designers to validate the readiness and effectiveness of their business processes before put them in operation. The presented algorithms are applicable to workflows with most common structural constructs, such as sequence, concurrency, and competition. A software tool was also developed to support automated analysis of the resource requirements and timing parameters of emergency response. A toy example was given to show the modeling and analysis of ROTWNs.

References

[1] N. R. Adam, V. Atluri and W. Huang, "Modeling and Analysis of Workflows Using Petri Nets", *Journal of Intelligent Information Systems*, pp. 131-158, March 1998.

[2] W.M.P. van der Aalst, "Three Good Reasons for Using a Petri Net-Based Workflow Management System", *Proceedings of the International Working Conference on Information and Process Integration in Enterprises* (IPIC'96), pp. 179–201, Nov 1996.

[3] M. Brachner, A simulation model to evaluate an emergency response system for offshore helicopter ditches, Proceedings of the 2015 Winter Simulation Conference, 2366-2377, 2015, CA, USA.

[4] J. Clempner, "Classical workflow nets and workflow nets with reset arcs: using Lyapunov stability for soundness verification," Journal of Experimental & Theoretical Artificial Intelligence, 29(1), p.43-57.

[5] S. Danial, F. Khan, B. Veitch, A Generalized Stochastic Petri Net model of route learning for emergency egress situations, Engineering Applications of Artificial Intelligence, 72, 170–182, 2018.

[6] Y. Deng, J. Wang, and J. Tsai, Formal analysis of software security architectures, *Proceedings 5th International Symposium on Autonomous Decentralized Systems*, Dallas, 426-434, 2001.

[7] Y. Deng, J. Wang, K. Beznosov and J. Tsai, "An approach for modeling and analysis of security system architectures," IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 5, 1099-1119, 2003.

[8] A. Filippoupolitis, L. Hey, G. Loukas, E. Gelenbe and S. Timotheou .Emergency Response Simulation Using Wireless Sensor Networks, Ambi-sys'08, 2008, Quebec, Canada.

[9] R. Hamzi, F. Innal, M. A.Bouda, M. Chati, Performance Assessment of an Emergency Plan Using Petri Nets, Chemical Engineering Transactions, 32, 235-240, 2013.

[10] K. van Hee, N. Sidorova and M. Voorhoeve, "Resource-constrained workflow nets," Fundamenta Informaticae, 71(2-3):243-257, 2005.

[11] G. Juh´as, I. Kazlov, and A. Juh´asov´a. Instance Deadlock: A Mistery behind Frozen Programs. PETRI NETS 2010, LNCS vol. 6128, pp. 1-17. Springer, 2010.

[12] M. A. Marsan and G. Chiola. On Petri nets with deterministic and exponentially distributed firing times. In: Advances in Petri Nets 1987, pages 132–145. Springer Berlin Heidelberg, 1987.

[13] M. Martos-Salgado, F. Rosa-Velardo: Dynamic Soundness in Resource-Constrained Workflow Nets. FMOODS/FORTE 2011: 259-273.

[14] T. Murata, Petri nets: Properties, analysis and applications, *Proceedings of the IEEE*, 77(4):541-580, 1989.

[15] M. Stoute, J. Wang, D. Rosca, W. Tepfenhart, and A. Milewski, "Workflow management tool support for incident command systems", *Proceedings of IEEE International Conference on Networking, Sensing and Control*, Fort Lauderdale, Florida, April 23-25,2006

[16] J. Wang, *Real-time Embedded Systems*, Wiley, ISBN: 978-1-118-11617-3, July 2017

[17] J. Wang, Y. Deng and C. Jin, "Performance analysis of traffic control systems based upon stochastic timed Petri net models," *International Journal of Software Engineering and Knowledge Engineering*. 10(6), 735-757, 2000.

[18] J. Wang and D. Li, "Resource oriented workflow nets and workflow resource requirement analysis," *International Journal on Software Engineering and Knowledge Engineering*, vol. 23, no. 5, 667-693, 2013.

[19] J. Wang, D. Rosca, W. Tepfenhart, and A. Milewski, "Incident command systems workflow modeling and analysis: A case study," *Proceedings of the 3rd International ISCRAM Conference* (B. Van de Walle and M. Turoff, eds.), Newark, NJ, May 2006.

[20] J. Wang and W. Tepfenhart, Formal methods in computer science, CRC Press, 2019.

[21] J. Wang, D. Rosca, W. Tepfenhart, A. Milewski, M. Stoute, An intuitive formal approach to dynamic workflow modeling and analysis, *Proceedings of International Conference on Business Process Management*, Nancy, France, 137-152, 2005.

[22] J. Wang, J. Tian and R. Sun, "Emergency healthcare resource requirement analysis: a stochastic timed Petri net approach," *IEEE International Conference on Network, Sensing and Control*, Zhuhai, China, March 30 - April 2, 2018.

[23] J. Wang, Jun Wang, G. Liu, and Q. Zeng, "Formal analysis of emergency department staffing based on stochastic timed Petri net models," *15th IFAC International Workshop on Discrete Event Systems*, Brazil, Nov. 2020.

[24] J. Zhou, J. Wang, and Jun Wang, "A simulation engine for stochastic timed Petri nets and application to emergency healthcare systems," *IEEE/CAA Journal of Automatica Sinica*, 6(4):969-980, July 2019.