International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:4, No:7, 2010

Dynamic-Stochastic Influence Diagrams: *Integrating Time-Slices IDs and Discrete Event Systems Modeling*

Xin Zhao, Yin-fan Zhu, Wei-ping Wang, and Qun Li

***Abstract*—**The Influence Diagrams (IDs) is a kind of Probabilistic Belief Networks for graphic modeling. The usage of IDs can improve the communication among field experts, modelers, and decision makers, by showing the issue frame discussed from a high-level point of view. This paper enhances the Time-Sliced Influence Diagrams (TSIDs, or called Dynamic IDs) based formalism from a Discrete Event Systems Modeling and Simulation (DES M&S) perspective, for Exploring Analysis (EA) modeling. The enhancements enable a modeler to specify times occurred of endogenous events dynamically with stochastic sampling as model running and to describe the inter-influences among them with variable nodes in a dynamic situation that the existing TSIDs fails to capture. The new class of model is named Dynamic-Stochastic Influence Diagrams (DSIDs). The paper includes a description of the modeling formalism and the hiberarchy simulators implementing its simulation algorithm, and shows a case study to illustrate its enhancements.

***Keywords*—**time-sliced influence diagrams; discrete event systems; dynamic-stochastic influence diagrams; modeling formalism; simulation algorithm

I. Introduction

THE weapon and equipment system of systems (SoS) demonstration is a high level complex systems analysis issue and involves vast uncertain factors. In the modern complex world, decision makers are faced with an ever increasing number of situations that require a all-around response, ranging from terrorisms to local battles and perhaps campaign between multi-national organizations for the employment of military forces to achieve or maintain demonstration objectives. To handle them effectively, Rand has gave a Exploring Analysis (EA)[1] methodology for Capabilities-Based Planning (CBP)[2][3]. As a model-based method, EA needs suitable model form for analysis supporting. The model's characters include: lower resolution; reflecting cause and effect relationship clearly; running promptly; less parameters; and could covering enough issue scope.

Xin Zhao is with the Institute of Systems Engineering, College of Information Systems and Management, National University of Defense Technology (NUDT), Changsha, Hunan, 410073, China (e-mail: zhx20mg@163.com).

Yi-Fan Zhu is with the College of Information Systems and Management, NUDT, Changsha, Hunan, China (e-mail: nudtzyf@hotmail.com).

Wei-ping Wang is with the National University of Defense Technology, Changsha, Hunan, China (e-mail: wangwp@nudt.edu.cn).

Qun Li is with the College of Information Systems and Management, NUDT, Changsha, Hunan, China (e-mail: liqun@nudt.edu.cn).

The extended Time-Sliced Influence Diagrams (TSIDs) [5][6] allows the existence of dynamic feedback loop circle to classic IDs[7] for graphic modeling. It's a special instance of Dynamic Bayesian Networks (DBNs) [8][9], that increases the decision nodes and the objective nodes for optimization of decision-making, not just probabilistic evaluation only. The TSIDs allows a system analyst to observe how uncertainties of variables influence model's behavior with time changing. TSIDs had been experimentally used in the area of EA for describing important factors of campaign and their relationships dynamically [10][11]. The TSIDs-based formalism is appropriate to answer influences queries and to perform what-if analyses.

Despite its ability to model complex situations in a compact and easy to read manner, TSIDs fails to capture variance of time sequences among events of dynamic situations. Furthermore, from another view, the object that TSIDs could describe is usual an independent process or subpart of the high-level process. This is its inherent deficiency for complex cooperated processes description.

In other words, although TSIDs connects 'uncertainty' with 'time' by a time-slice concept, it still can't be fit for all kinds of high level modeling. If the 'system' not only has complex entities but also complex relationships, it's not easy to reflect its process clear. Take military campaign for example, if it contains interactional crossed loops that caused by different weapons and troops, it's hard to abstract them.

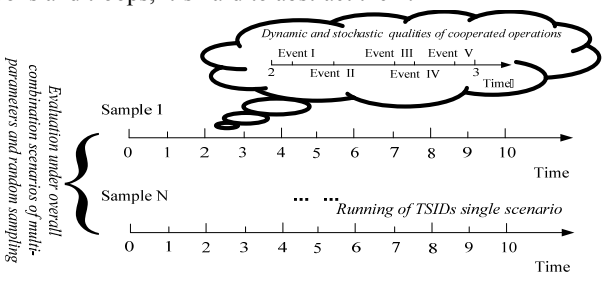

Fig. 1 Temporal Sequence of TSIDs

Moreover, the relationship description between times or influence factors are settled in TSIDs. It's not enough to describe the dynamic and stochastic cooperated operations. As Fig.1, TSIDs just runs short of variational temporal sequences among different time-slices.

This shortcoming of TSIDs may turn out to be unrealistic in

many real world situations. The paper proposes syntax and semantic enhancement to TSIDs to overcome the limitation. The enhancement would enable a system analyst to model the impacts of different actions on the desired effect in a dynamic uncertain situation.

The rest of the paper is organized as follows. Section II describes the technical background-IDs, TSIDs, and the combination for TSIDs and DES. Sections III and IV describe the proposed modeling formalism and its simulation algorithm, respectively. Section V illustrates the proposed enhancements by a case study that realized in EASim-an IDs modeling tool. Finally, Section VI concludes the paper and points towards the future research direction.

II. Technical Background

A. Classic IDs

Classic Influence Diagrams (IDs) [7] is a directional acyclic graph for decision-making issue modeling. It's also a probability net being used for solving uncertainty and reflecting qualitative & quantitative knowledge synthetically and an effective tool for information communication. Decision-making analysis based on IDs is usual at strategic level. The classic IDs is formed by directional arcs and nodes with data structures, as Fig.2.

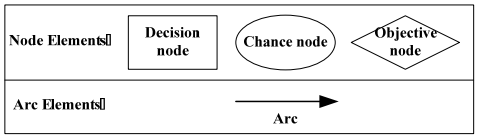

Fig. 2 Graphical elements of classic IDs

The developing process of IDs is field and application oriented basically. Their evolvement separately or simultaneously occurred at relation, function and value level, including[12][13][14][15][16][17][18]: Probability IDs, Fuzzy IDs, Rough IDs, Potential IDs, Monotonic IDs and Ignorant IDs etc. No matter what requirements of application they fulfilled or what levels they occurred at, the essential purpose is to utilize the IDs' powerful abilities to graphically capture causal and influence relationships for analytical modeling that is a kind of methodology for analyzing complex strategic decision-making issue. Among them, Time-Sliced IDs (TSIDs) is especially worthy to get attention for model-based high level issue decision-making analysis.

B. Time-Sliced IDs

By introducing time concept and dynamic character, TSIDs extended classic IDs and could support differential equation and continue system modeling for analysis. This also made it more fit for abstracting averaged process, and improved cause and effect influence net's description comparing with static analytic IDs, as Fig.3.

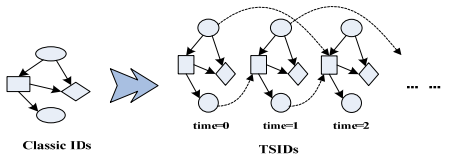

Fig. 3 Classic IDs to TSIDs

An TSIDs is a four-tuple $TSIDs =< D, M, A, h_N >$ where

- D is the index set of nodes;
- $M = \{M_d \mid d \in D\}$ is the set of variable nodes;
- A is the set of influence arcs;
- h_N is the iterative step.

C. Combination of TSIDs and DES M&S

The capability to enhance that TSIDs needs is just what the Discrete Event Systems (DES) M&S owned. DES [19][20][21] can express the input and intrinsic random, time delay, and queue. And so it can describe simultaneous, cooperating and multilevel complex processes. It's a good reference for TSIDs improvement. The combined modeling of TSIDs and DES can form a new formalism to enhance the top-down complex system modeling. So, when you do a high level complex issue modeling, the TSIDs part can be used to model the issue's macrostructure parts well, and the DES part can be used to model other detailed parts.

1) Relationship of 'time' conception: The two parts (TSIDs and DES) should have accordant 'time' conception. So 'Etime' is introduecd to represent event time comparing with 'Ftime' that representing feedback time, as Fig.4.

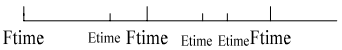

Fig. 4 Event time and feedback time

2) Combination mode: From the graphic syntax and semantic point of view, the nodes in DES M&S must have bidirectional relationship with the variables of TSIDs, whether events, states, activities, or processes they figured. Considering discrepancies of two parts, their elements may have two kinds of combination modes: one is node-combined, what need holistic simulated evaluation, and is a multi-granularity description to the same level and towards behavior (event) principally; the other is module-combined, what need separated simulated evaluation, and is a multi-resolution description to different levels and towards entity principally. The former mode is flexible and suit for complex system description especially to our situation.

3) Principles: As amelioration to existing method, it just should

- Be graphic, and needs less extension to TSIDs;
- Be intuitionist;
- Be prone to understand, and be convenient for model structure redress and relationship explanation;
- Unitive rule, and be convenient for software design and implementation.

III. Proposed Modeling Formalism

In this paper, the improved method is called as 'Dynamic-Stochastic Influence Diagrams (DSIDs)', for its ability for dynamic arranging of stochastic events, processes and influences, and supporting their random treating. To describe the characters of DES, DSIDs add a 'Time node' at TSIDs foundation, expressing stochastic events by time list directly, as Fig.5. This is propitious to assort with nodes of TSIDs, and to transfer evaluating engine.

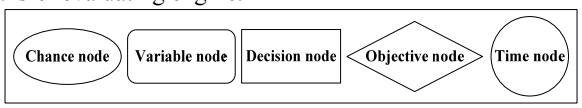

Fig. 5 Primary variable nodes of DSIDs

The DSIDs describes pivotal elements of issue by these graphic nodes, and the connection of them form a DSIDs net model. In addition, as a graphic modeling method, it should be considered that doing some structural processing for input and output interfaces during the description of DSIDs formalism. A standard DSIDs model specification is a structure:

$$DSIDs =< D, M, A, I, Z, h_N, t, ta >$$

Where

- D is the index set of nodes;
- $M = \{M_d \mid d \in D\}$ is the set of variable nodes;
- A is the set of influence arcs;
- $I = \{I_d \mid d \in D\}$ is the set of nodes that influence node M_d;
- $Z = \{Z_d \mid d \in D\}$ is the interface mapping set of node M_d;
- h_N is the iterative step;
- t is the current time;
- ta is the time advance function.

A. Nodes Set

There are five kinds of nodes in DSIDs, $M = N_C \cup N_V \cup N_D \cup N_O \cup N_T$ namely chance node, variable node, decision node, objective node, and time node. Hereinto, the variable form of time node is chain list for value storage representing the time that event will take place.

To $\forall d \in D$, $M_d =< n, X_d, Y_d, V_{init}, V_d, \Delta_d, \Lambda_d, t, h >$, in which

- $n \in NP$ is the name of node, and NP is the name space;
- t is the current time;
- h is the current actual step;
- V_{init} is the initialized value, maybe a number or a depending function to other nodes;
- V_d is the value of node to each time;
- $X_d = \{(p, v) \mid p \in IPorts_d, v \in X_{p_d}\}$ is the multivariable set of inputs of the node with $IPorts_d$;
- $Y_d = \{(p, v) \mid p \in OPorts_d, v \in Y_{p_d}\})$ is the multivariable set of outputs of the node with $OPorts_d$;
- Δ_d is the internal logic function;
- Λ_d is the output mapping function.

Of course, to different nodes, the detailed definitions are different that be omitted for conciseness here.

The semantic executing of node can described as follow: After computed the time advanced by node net (DSIDs), nodes receive the value of t and h; when received the value $X_{d(t)}$ from all input ports, based on themselves' forepassed value, them transact internal logic by Δ_d, calculate output values by Λ_d and map them to output ports.

B. Arcs Set

The influence relationships are figured by arcs in DSIDs and different for the sort of nodes they connected, to form the net structure together with all nodes. In DSIDs, $A = A_R \cup A_I \cup A_E \cup A_F \cup A_C \cup A_A$, as Tab.I.

The former four are the same as those of TSIDs, and the later

TABLE I
Arrows and their signification

Arc	Definition	Signification
A_R	$\{(x, y), x \in N_C \cup N_V, y \in N_V \cup N_O\}$	relevancy
A_I	$\{(x, y), x \in N_C \cup N_V, y \in N_D\}$	information
A_E	$\{(x, y), x \in N_D, y \in N_V \cup N_O\}$	influence
A_F	$\{(x, y), x \in N_D, y \in N_D\}$	order
A_C	$\{(x, y), x \in N_V, y \in N_T\}$	condition
A_A	$\{(x, y), x \in N_T, y \in N_V \cup N_T\}$	action

two are what DSIDs has peculiarly. Respectively, A_C indicates condition that its source node provides to target node, and A_A indicates action that its source node affects to target node when the event happened.

In addition, there is also a loop restriction in DSIDs as follow: In the net connected by A_R, A_I, A_E, A_F and correlative nodes, there couldn't exist circle depending that without time delay. Namely, to $\forall i, j \in D$, if the calculation of $V_{i(t)}$ indirectly depends on $V_{j(t)}$, then the calculation of $V_{j(t)}$ couldn't depends on $V_{i(t)}$ synchronously, no matter directly or indirectly.

C. Other Constituents

1) The set of nodes that influence M_d : To $\forall d \in D : I_d = \{i \mid ((i, fromport), (d, toport)) \in IC\}$,where $IC \subseteq \{((a, op_a), (b, ip_b)) \mid a, b \in D, op_a \in OPorts_a, ip_b \in IPorts_b\}$ that expressing linked relationships.

2) The interface mapping set of M_d : To $\forall d \in D : Z_d = \underset{i \in I_d}{\times} Y_i \to X_d$, $Z_d(..., y_{(i, fromport)}, ...) = x_{(d, toport)}$, and

$$x_{(d, toport)} = \underset{\{(i, fromport) \mid ((i, fromport), (d, toport)) \in IC\}}{\oplus} y_{(i, fromport)}$$

Where $\oplus x = x$; if $x_i \neq \varnothing \wedge \forall j \neq i \Rightarrow x_j = \varnothing$, then $\oplus x_1, x_2, ..., x_n = x_i$, else $\oplus x_1, x_2, ..., x_n = \varnothing$. ' $\oplus$ ' allow

several output ports linked to one input port. It's helpful to DES expression [21].

3) t and $h_N : t$ is the current time, h_N corresponds to the 'Ftime' conception that mentioned before, they are calculated by function '*ta*'.

4) Time advance function 'ta': The time-advance of DSIDs needs to consider time values updating and events arising. This situation is similar to multiformalism M&S of discrete time and discrete event systems. There is a similar concept 'state event[21]' too. The implementation of function *ta* followed the 'bisection method' arithmetic that the article [21] mentioned. It's the function of t, h_N and V_d of all nodes, and so

$$ta : \underset{d \in D}{\times} V_{d(t)} \times t \times h_N \to R^{+}_{0,\infty}.$$

IV. Simulation Algorithm for DSIDs Evaluation

The algorithm of DSIDs simulated evaluation could be described as Fig.6. To see the detail of TSIDs evaluation algorithm implementation, you may take [5][6][9][22] for references. The algorithm and the 'forward calculate' will be materialized in the description of simulators next.

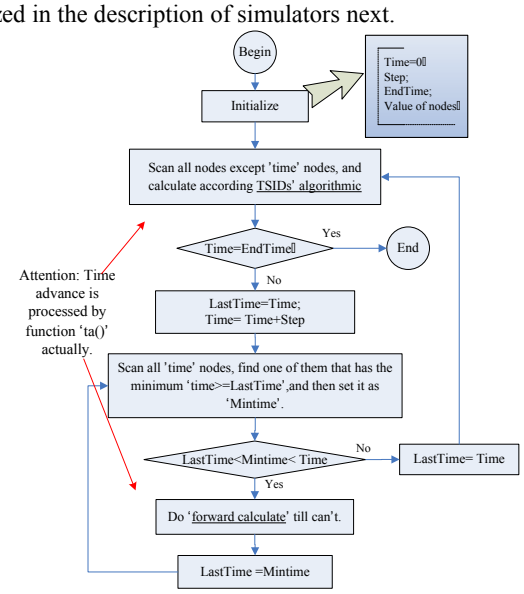

Fig. 6 The logistic process of DSIDs evaluation algorithm

The 'LastTime' of algorithm expresses the latest calculating time that between 'Time' and 'Time-Step' (Step=1 here), as Fig.7.

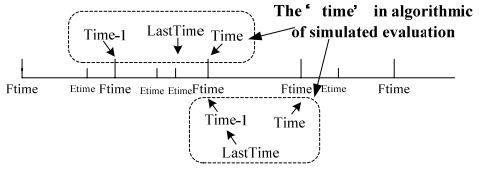

Fig. 7 The 'time' in evaluation processing

The algorithm is compatible for TSIDs, and so they (TSIDs & DSIDs) are equivalent without 'time node'.

A. Hiberarchy

The simulators of DSIDs may have clear hiberarchy as Fig.8. The DSIDs simulator structure is compartmentalized into 'node-simulator' for 'node' and 'net-coordinator' for 'net' that formed by nodes. The former control the operation of nodes, and the later define the connection of them. Moreover, 'net-coordinator' also takes charge for time advancing, state events scheduling, and messages transferring etc. In addition, a 'root-simulator' is introduced to control simulation repetition which will not be discussed detailed in this paper.

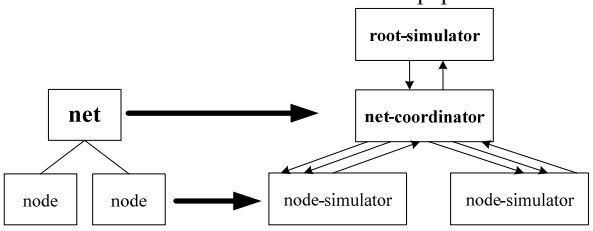

Fig. 8 Hiberarchy of DSIDs simulators

The simulation algorithm of DSIDs is implemented based on the following seven kinds of messages. DSIDs scheduling process is realized by messages communication.

TABLE II
Seven Kinds of Messages in Communication

Name	Message	Send and Receive
i	Initialization	upper to lower
*	Calculation enabled	net-coordinator to node-simulator
e	Event schedule	net-coordinator to node-simulator of N_T
r	Earliest time of the future event	node-simulator of N_T to net-coordinator
se	Failure of state event	node-simulator of N_T to net-coordinator
x	Value input	net-coordinator to node-simulator
y	Value output	node-simulator to net-coordinator

Tab.II shows symbols of seven messages on which the DSIDs evaluation is based, while their sending and receiving mode are described in Tab.III.

TABLE III
Massage Send and Receive for Each Simulator

Message	node-simulator					net-coordinator	root-simulator
	N_C	N_V	N_D	N_O	N_T		
(*i*, *t*)	↓	↓	↓	↓	↓	↓→	→
(*, *t*)	↓	↓	↓	↓	↓	↓→	--
(*e*, *t*)	--	--	--	--	↓	→	--
(*r*, *t*)	--	--	--	--	→	↓	--
(*se*, *t*)	--	--	--	--	→	↓	--
(*x*, *t*)	--	↓	↓	↓	↓	→	--
(y, t)	→	→	→	--	→	↓	--

Legends: ↓ : receive; → : send; --: void.

B. Description of Simulators

The following sections will show detail of algorithm realization for each simulator.

1) Node-Simulator: There are four node-simulators for different kinds of nodes. They all base on the basic

node-simulator and have each peculiarity. Fig.9 to Fig.14 shows details.

```
basic - node - simulator for M_d
  variabl es :
     parent   // net - coordinato r
     tl       // time of last
     tn       // time of now
     M_d =< n,X_d,Y_d,V_init,V_d,Δ_d,Λ_d,t,h >
  when receive i - message (i,t) at time t
     tl = t
     tn = t
  when receive * - message (*,t) at time t
     if t ≤ tn then
        error : bad synchroniz ation
  when receive e - message (e,t) at time t
     if t ≤ tn then
        error : bad synchroniz ation
  when receive x - message (x,t) at time t
   //wi th input valu e x_(r,toport) =
   //        Z_(d*,frompo rt),(r,top ort) (y_(d*,frompo rt))
     if t ≤ tn then
        error : bad synchroniz ation
 end basic - node - simulator
```

Fig. 9 Modality of basic node-simulator

```
node - simulator for n ∈ N_C
  when receive * - message (*,t) at time t
     if Y_d = ∅ then
        error : bad synchroniz ation
     Y_d = V_d(t) = Δ_d (t,h)
     send y - message (y,t) to parent
     tl = tn
     tn = t
 end node - simulator
```

Fig. 10 Node-simulator for chance node

```
node - simulator for n ∈ N_D
  when receive * - message (*,t) at time t
     if Y_d = ∅ then
        error : bad synchroniz ation
     if X_d = ∅ then
        Y_d = V_d(t) = Δ_d (t,h,V_d)
        send y - message (Y_d,t) to parent
        tl = tn
        tn = t
  when receive x - message (x,t) at time t
     x_(d,toport)(t) = x_(r,toport)
     if Y_d = ∅ then
        error : bad synchroniz ation
     if X_d = ∅ then
        error : bad synchroniz ation
     if ∃x_(d,port)(t) = ∅ then //  x_(d,port) ∈ X_d
        do nothing
     Y_d = V_d(t) = Δ_d (t,h,V_d,X_d(t))
     send y - message (Y_d,t) to parent
    tl = tn
    tn = t
 end node - simulator
```

Fig. 11 Node-simulator for decision node

```
node - simulator for n ∈ N_V
  when receive x - message (x,t) at time t
     if Y_d = ∅ then
        error : bad synchroniz ation
     if X_d = ∅ then
        error : bad synchroniz ation
     x_(d,toport)(t) = x_(r,toport)
     if ∃x_(d,port)(t) = ∅ then //  x_(d,port) ∈ X_d
        do nothing
     Y_d = V_d(t) = Δ_d (t,h,V_d,X_d(t))
     send y - message( Y_d,t) to parent
    tl = tn
    tn = t
 end node - simulator
```

Fig. 12 Node-simulator for variable node

```
node - simulator for n ∈ N_O
  when receive x - message (x,t) at time t
     if Y_d ! = ∅ then
        error : bad synchroniz ation
     if X_d = ∅ then
        error : bad synchroniz ation
     x_(d,toport)(t) = x_(r,toport)
     if ∃x_(d,port)(t) = ∅ then //  x_(d,port) ∈ X_d
        do nothing
     V_d(t) = Δ_d (t,h,V_d,X_d(t))
    tl = tn
    tn = t
 end node - simulator
```

Fig. 13 Node-simulator for objective node

```
node - simulator for n ∈ N_T
  variabl es :
     tle    // t ime of last event
     ar_int // accurary reached judgement function
     P      // priority
     cancel_int // cancel event
  when receive e - message (e,t) at time t
     if Y_d = ∅ or X_d = ∅ then
        error : bad synchroniz ation
     if ∃x_(d*,port)(t) = ∅ then
                  // x_(d*,port) ∈ X_d ∧ d* ∉ D_N_T
        error : bad synchroniz ation
     find earliest t ime t_e from V_d //just find it
     if t_e > t then
        send r - message( t_e,t) to parent
        tl = tn
        tn = t
     if Δ_d (t,h,X_d(t)) = false then //e vent conditions
        send r - message( t_e,t) to parent
        tl = tn
        tn = t
     if ar_int (t_e,t) = false then
        send se - message( se,t ) to parent
    tak e t_e out from V_d          //ta ke it out
     if cancel_int (t_e,V_d,t,h,X_d(t)) = false then
        tle = t
        tl = tn
        tn = t
        h = tn − tl
        Y_d = ∧_d (tle,t,h,X_d(t))
        send y - message( Y_d,t) to parent
        find earliest t ime t_e from V_d //just find it
        send r - message( t_e,t) to parent
  when receive x - message (x,t) at time t
     if d* ∉ D_N_T then
        x_(d,toport)(t) = x_(r,toport)
     add x_(r,toport) to V_d
 end node - simulator
```

Fig. 14 Node-simulator for time node

What needs to be specially explained is that the simulator for time node: differentiate input values that from N_T and non- N_T ; includes two special functions, $ar_{\text{int}}(t_e,t)$ for state event accuracy judgment and $cancel_{\text{int}}(t_e,V_d,t,h,X_{d(t)})$ for event canceling; and owns several output ports to drive other time nodes or influence linked variable nodes.

2) Net-Coordinator: After the definition of node-simulators, the net-coordinator, whose task is to send and receive messages to make time advanced and nodes calculated, could be discussed. See Fig.15 for details.

```
net - coordinato r
  variabl es :
     parent     // root - coordinato r
     tl         // time of last
     tn         // time of now
     tf     // time of future (next step time)
     fTime      // FTime
     DSIDs  =< D,M,A,h N ,ta,I,Z >  //net
  when receive i - message (i,t ) at time t
     for each d in D do
        send i - message( i,t ) to child d
    fTi me = t
     tl = t
     tn = t
     tf = +∞
     send * - message( *,t ) to self
  when receive * - message (*,t )
    whi le t = fTime + h N do fTime = t
     tl = tn
     tn = t
     tf = +∞
     for each d in D do
        send * - message( *,t ) to child d
     loop
        for each d in D N T do
           send e - message( e,t ) to child d
                //accord ing P : priority
     until there wasn' t event occurred
     if tf > fTime + h N then
        tf = fTime + h N
        send * - message( *,tf ) to self
     send * - message( *,tf ) to self
  when receive r - message (r,t ) at time t
     if tn < r < tf then
        tf = r
  when receive se - message (se,t ) at time t
     stop the computatio n of all children
     rollback all variables and events to time = tl
     tf = (tf-tn)/ 2   // ta algorithm
     send * -message( *,tf ) to self
  when receive y - message (y,t ) from d*
       //with output val ue y (d*,frompo rt)
     receivers = {(r,toport ) | r ∈ D}
     / /d* ∈ I r ,Z (d*,frompo rt),(r,top ort) (y (d*,frompo rt) ) ≠ ∅
     for each (r,toport ) in receivers
       send x - messages (x (r,toport) ,t ) to (r,toport )
        //with input valu e
        // x (r,toport) = Z (d*,frompo rt),(r,top ort) (y (d*,frompo rt) )
end net - coordinato r
```

Fig. 15 Net-coordinator

V.A Modeling Tool and Case Study

A. EASim - Modeling Tool for DSIDs

A software tool – EASim, was designed and implemented, as Fig.16, which provides the ability and agility to deal with complex issues of real world, by supporting DSIDs further for analytical modeling.

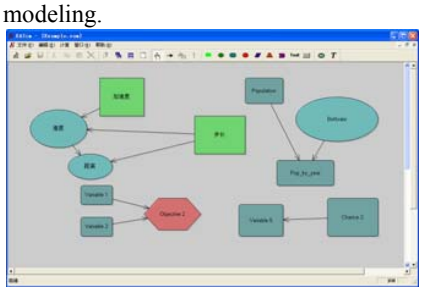

Fig. 16 EASim-software tool for DSIDs modeling

In EASim,

- The hiberarchy description makes it more simplified to build or manage big model and increases the ability for analytical model's building, organizing and evaluating.
- And variables can also be vector array except scalar. It's another source of flexible and abstract ability for model maker. It will redound to make some treat-off among model particular, evaluating time, available data and dimensionality space.
- To improve representation of discrete processes, EASim not only samples at each evaluating, but also samples at every steps in each evaluating, and provides several statistical methods for analyzing.
- A script language, Python, is embedded in EASim; and extended by C++ to provide more power for analytical modeling.

In addition, the DSIDs evaluation may include vast data and complex process, especially for complex military SoS counterwork. So, EASim's evaluating engine is not to immit all correlative data of the model into memory, but to storage model using XML format files, and then respectively evaluate these files by pretreating them to multi-batches. It not only improves running efficiency of evaluating, but also relaxes strict memory and computing power requirements to PC – analysis is often carried out in a combinatorial space of all factors' various values and common needs to be performed just by PC.

B. DSIDs Model of Armada Area Defense Issue

An armada area defense issue is taken for DSIDs modeling example. The basic scenario is that:

- Attacking planes, from different batches, pierce through the defense lines and then fires cruise missiles to attack weapon carriers of defending side for anti-ship missile intercepting;
- Attacking side fires some ballistic missiles to attack kernel target directly at opportune time;
- Defending side intercepts the attacking planes and missiles by air-to-air battles and air defense missiles.

Fig.17 shows the macrocosm structure of the DSIDs model and the influences of factors between time nodes and other nodes, for armada area defense issue.

Fig.18 shows the gross num vs. useable num of node

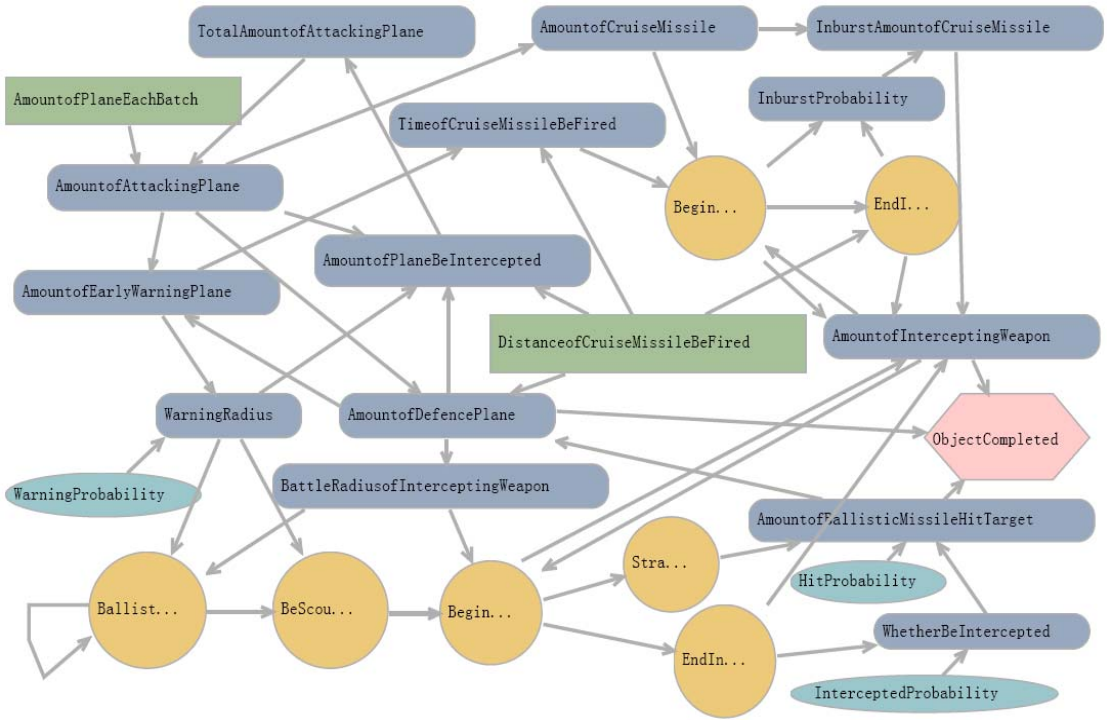

Fig. 17 A reduced version of DSIDs model for armada area defense

'AmountofInterceptingWeapon' in the DSIDs model. It illustrates that the competition of war resources could came forth at any moment and this could influence the actual occurrences of events.

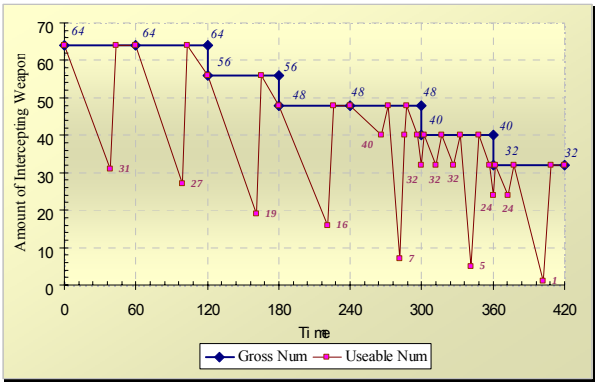

Fig. 18 Gross vs. useable amount of missile intercepting weapons

Fig.19 shows the occurred time of typical events in the DSIDs model. It illustrates that the campaign processes and events them contained could be advanced well, and exhibits the influences that DSIDs could described.

VI. Conclusion

Classic IDs is a good tool to support decision-making for its laconic graphics and effectual approximate reasoning. And the extended Time-sliced IDs has be used for strategic planning successfully. To get more creditability for special object, the weapon and equipment system of systems (SoS) demonstration, one of strategic planning issues, the shortcoming of TSIDs is discussed and has been enhanced to own more capability to describe more details of battle processes for influence factors analysis. The proposed class of model is named Dynamic-Stochastic Influence Diagrams (DSIDs). And its modeling formalism and simulation algorithm was described amply. It had been used in the martial planning example excellently. To make the new formalism could be used widely, more work have to do further. For example, as a combination of TSIDs and DES M&S, DSIDs has relationships with DBNs, DBNs, colored Petri nets, and Event Graphs etc, the conversions to these interrelated systems make it's possible to use a variety of analysis algorithms developed for them.

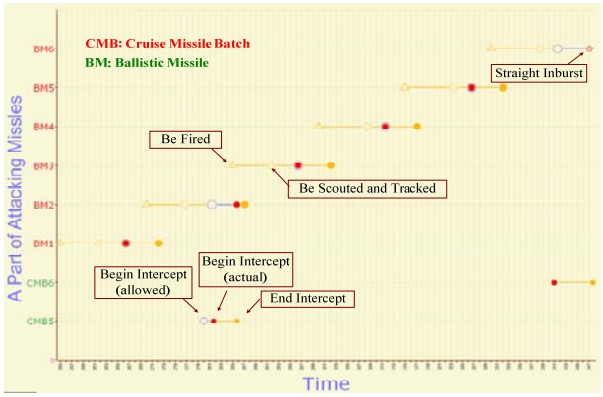

Fig. 19 Occurred time of typical events in the DSIDs model

ACKNOWLEDGMENT

The authors would like to gratefully acknowledge the financial support of the National Defense Pre-Research Foundation of China (Grant No. 9140C640505), the National Natural Science Foundation of China (No. 60974073 and No. 60974074).

REFERENCES

[1] Paul K. Davis, "Exploratory Analysis Enabled by Multiresolution, Multiperspective Modeling", In Proceedings of the 2000 Winter Simulation Conference.
[2] Paul K. Davis, "New Paradigms and New Challenges", In Proceedings of the 2005 Winter Simulation Conference.
[3] Paul K. Davis, "Lessons from Defense Planning and Analysis for Thinking About Systems of Systems", Prepared for the Symposium on Complex System Engineering, Santa Monica, Calif.: RAND Corporation, 2007.
[4] Paul K. Davis, "Amy Henninger. Analysis, Analysis Practices, and Implications for Modeling and Simulation", Santa Monica, Calif.: RAND Corporation, 2007. pp. 5-6.
[5] Analytica User Guide, Lumina Decision Systems, Inc.
[6] M. Granger Morgan and Max Henrion, "Chapter 10 of Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis", Cambridge University Press, New York, 1990, reprinted in 1998.
[7] R.A. Howard, J.E. Matheson, "The Principles and Applications of Decision Analysis", vol. II, Strategic Decisions Group, Menlo Park, CA, 1984, pp. 720–762 (Chapter: influence diagrams).
[8] Figueroa, G. A., and Sucar, L. E., "A Temporal Bayesian Network for Diagnosis and Prediction", In Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, 1999.
[9] Murphy, K., "Dynamic Bayesian Networks: Representation", Inference and Learning, PhD Thesis, UC Berkley, Jul. 2002.
[10] P.K, D., J.H. Bigelow, and J. McEver, "EXHALT: An Interdiction Model for Exploring Halt Capabilities in a Large Scenario Space", Vol. MR-1137- OSD. 2000, Santa Monica, CA: RAND.
[11] Davis, P.K., J.H. Bigelow, and J. McEver, "Exploratory Analysis and a Case History of Multiresolution, Multiperspective Modeling", reprint volume RP-925. 2001, Santa Monica: RAND.
[12] Miguel López-Díaz, Luis J, "Rodríguez-Muñiz. Influence Diagrams with Super Value Nodes Involving Imprecise Information", European Journal of Operational Research (S0377-2217), 2007, 179: 203-219.
[13] Michael Diehl, Yacov Y, "Haimes. Influence Diagrams with Multiple Objectives and Tradeoff Analysis ", IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans (S1083-4427), Vol. 34, NO. 3, May 2004:293-304.
[14] R.D. Shachter, M.A. Peot, "Decision Making Using Probabilistic Inference Methods", in: Proceedings of the 8th Conference on Uncertainty in Artificial Intelligence, San Jose, 1992, pp. 276-283.
[15] R.D. Shachter, P.P Ndilikilikesha, "Using Potential Influence Diagrams for Probabilistic Inference and Decision Making", in: Proceedings of the 9th Conference on Uncertainty and Artificial Intelligence, 1993, pp. 383-390.
[16] P.P. Ndilikilikesha, "Potential Influence Diagrams", International Journal of Approximate Reasoning 11 (1994) 251-285.
[17] Koller D, Milch B, "Multi-agent Influence Diagrams for Representing and Solving Games". IJCAI. Seattle,USA: Elsevier. 2001: 1024-1034.
[18] N.L. Zhang, "Probabilistic Inference in Influence Diagrams", in: Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Francisco, 1998, pp. 514-522.
[19] Paul A.Fishwick, "Simulation Model Design and Execution", Prentice-Hall Inc, 1995.
[20] Eric L. Savage, Lee W. Schruben, and Enver Yücesan, "On the Generality of Event-Graph Models", INFORMS Journal on Computing, Vol. 17, No. 1, Winter 2005, pp. 3– 9.
[21] Zeigler, B. P., T. G. Kim and H. Praehofer, "Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems", second edition Academic Press, 2000.
[22] ZHAO Xin, ZHANG Wei, LEI Yong-lin, ZHU Yi-fan, "Time-Sliced Influence Diagrams for Analytical Modeling", The 2nd IEEE International Conference on Advanced Computer Control, 27-29, March 2010, Shenyang, China.