https://doi.org/10.5392/IJoC.2018.14.3.007

A Simulation Method For Virtual Situations Through Seamless Integration Of Independent Events Via Autonomous And Independent Agents

Jong Hee Park
School of Electronics engineering, Graduate School, IT College
Kyungpook National University, Sangyuk-Dong, Daegu, Korea

Jun Seong Choi
School of Electronics engineering, Graduate School, IT College
Kyungpook National University, Sangyuk-Dong, Daegu, Korea

ABSTRACT

The extent and depth of the event plan determines the scope of pedagogical experience in situations and consequently the quality of immersive learning based on our simulated world. In contrast to planning in conventional narrative-based systems mainly pursuing dramatic interests, planning in virtual world-based pedagogical systems strive to provide realistic experiences in immersed situations. Instead of story plot comprising predetermined situations, our inter-event planning method aims at simulating diverse situations that each involve multiple events coupled via their associated agents' conditions and meaningful associations between events occurring in a background world. The specific techniques to realize our planning method include, two-phase planning based on inter-event search and intra-event decomposition (down to the animated action level); autonomous and independent agents to behave proactively with their own belief and planning capability; full-blown background world to be used as the comprehensive stage for all events to occur in; coupling events via realistic association types including deontic associations as well as conventional causality; separation of agents from event roles; temporal scheduling; and parallel and concurrent event progression mechanism. Combining all these techniques, diverse exogenous events can be derived and seamlessly (i.e., semantically meaningfully) integrated with the original event to form a wide scope of situations providing chances of abundant pedagogical experiences. For effective implementation of plan execution, we devise an execution scheme based on multiple priority queues, particularly to realize concurrent progression of many simultaneous events to simulate its corresponding reality. Specific execution mechanisms include modeling an action in terms of its component motions, adjustability of priority for agent across different events, and concurrent and parallel execution method for multiple actions and its expansion for multiple events.

Key words*: Inter-Event Planning, Immersive Learning, Autonomous Agent, Pedagogical Experience.*

1. INTRODUCTION

Virtual World (VW)-based pedagogical systems strive to provide realistic experiences in immersed environment [1]. Like ones for procedural knowledge, VW pedagogical systems for declarative knowledge such as mathematics and linguistics [2], [3] also have good reason to share implicate events as the effective learning stage, in that numerous opportunities of pedagogical experience could be immersively embedded in progression of those events [4], [5]. Instead of story plot comprising predesigned situations, we develop an inter-event planning method for simulating situations involving multiple events coupled via their associated agents' conditions and realistic associations between events. The extent and depth of the event plan determine the scope of pedagogical experience in situations and consequently the quality of learning in a language-learning Intelligent Tutoring System (ITS) based on our simulated world.

Within approaches for Interactive Storytelling (IS), artificial intelligent based IS is more specifically related to the mechanisms for automatic story generation [6]. An IS system that continuously adjusts the story based on the user's behaviors is often applied for achieving the author's desired effects in the story when dealing with a variety of user interactions [7]. Several techniques can be used to correspond to changing situations in IS systems, e.g., action repair which is dedicated to restoring executability conditions and/or reaching the same final state as the original action, or situated reasoning which essentially consists in interrupting the current plan and dealing with a specific situation arising [6]. These IS methods, like conventional story-based systems [3], [8], however, allow

* *Corresponding author, Email: daegulink@naver.com*
Manuscript received Sep. 01, 2017; revised Dec. 19, 2017; accepted Dec. 19, 2017

a limited range of narrative variations within the boundary of an individual event as pre-designed by an author.

The agents play pivotal roles in all intentional events as the primary constituents of the virtual situations. Non-player characters (NPCs) in IS have not usually been modeled as autonomous and independent agents as the player or lead character. Those supporting agents are likely to be designed to act at best only reactively, and their personal conditions or belief are little considered in planning. Our agents are modeled to behave proactively, that is, to be capable of planning ahead. Such an enhancement requires any agent to be a full-blown type with its own belief and planning capability regardless of its role.

To expand the chance of pedagogical experience, our planning scope is not to be confined within an individual event, but to span across many events a real situation usually comprises. Those events are to be meaningfully coupled via common factors between events or realistic inter-event associations. The inter-event association types include deontic associations as well as conventional causality [9]. As a result a main event is linked with exogenous events through its agents cast in its roles. All those coupling factors are designed as part of a common background world, which allows exogenous events to be derived and seamlessly (i.e., semantically meaningfully) integrated with the main event. Also, all the events are separated from the background world (including agents), which provide another clue for enhanced diversity of situations. Coupling between independently-planned events is based on an agent's multitude of roles across concurrent events. These events in a plan may progress concurrently or digress toward a newly-chosen main goal replacing the current goal or event, and the plan could be merged or fragmented according to their respective lead agents' intentions.

Our planning model chooses agents' actions as the primitive elements of events, that is, each action belongs to capability of an agent and is the animated unit of implementation. This fine level of primitive leads to a close integration among main event and exogenous events. It also enables new events to be efficiently executed in reaction to abrupt changes in relevant conditions, and allows for parallel execution of agent's actions. This parallelism in the unit of action provides uniformity in the unit of execution and action's independency from preceding body state, which affords animation scalability against the infinite diversity of changes occurring in the virtual world. Further, events in our planning are aligned along time line so as to be scheduled according to their exact timing beyond relative precedence.

2. RELATED RESEARCH

A number of planning methods have been proposed for interactive narrative systems [6], [10], [11]. Their inherent concern is balance between narrative variation and plot coherence. In HTN planning, high-level tasks are decomposed into simpler tasks until a sequence of primitive actions solving the high-level tasks is generated. Several techniques using HTN planning have been proposed (e.g., action repair or situated reasoning) to enhance narrative variation [12], but its scope is confined within the pre-authored boundary. A major difference of our event planning from HTN-based planning is that the primitive actions in our planning refer to innate physical capability of an entity (including agent) instead of playable actions. Furthermore, in addition to this 'vertical' decomposition, our inter-event planning model generatively searches the background world in a 'horizontal' direction for satisfying the required preconditions of events identified.

Behavior tree enables sophisticated sequences of actions and contingencies to be represented as a concise graphical structure following a set of very simple rules with equivalent representations [13]. Behavior Trees are appropriate as tools for dealing with the particular situations, but it is not appropriate for building an autonomous virtual world due to the necessity of extensive intervention of an author. While our model also expresses the story progression in terms of tree structure, it is decomposed into fragments to be assigned to different autonomous and independent agents and executed according to their order identified in planning.

A crucial premise for planning a situation consisting of a number of intertwined events is to provide a stage for those events to be meaningfully coupled in a spatio-temporal context. An integrated representation scheme is designed to depict not only present and past but future in an effective and intuitive manner [9]. The resulting multi-dimensional comprehensive knowledge structure provides the basis for a multi-layered virtual world to be used as the full-blown background world to contextualize events with. It is also used as the basis for implementing agent's belief and student's model.

By designing an action as a sequence of motions [14], transfer from an action to another action can be effectively realized regardless of the initial state or position of its body. As a result, actions composing events coupled via realistic associations are executed independently of results from their preceding actions. Parallel or concurrent actions based on assembling multiple actions are extended beyond multiple motions in order to simulate parallel and concurrent events without any event being unnecessarily interrupted.

While episodes in edugame are independent from each other, that is, isolated [8], events in our model, corresponding to episodes, are interconnected (at least loosely) via 'some' association, if not parts of one coherent story. Discovery learning systems [15] share with our pedagogical model the pedagogical paradigm of self-directed exploratory form of learning. Whereas a scientific discovery learning system in particular emphasizes the student's deep, conceptual understanding of its hard learning domain, our model is designed for shallow but broad-ranged subjects. Due to the dynamic generation of the story, the scope of learning in our model is not confined to (part of) a story plan [15] or to even each scene within [6], [10].

3. INTER-EVENT PLANNING

3.1 Modeling agent's action as primitive element of event

In our simulation model, an action refers to an agent's capability to act, and its actual occurrences are realized in terms of its iteration or continuation. An event refers to an activity

that involves multiple agents assuming their respective roles therein. Each such role is designated to perform one or more actions for the event. An event is eventually carried out by performing those actions required of those agents cast in its associated roles. Those actions are to be properly arranged into a plan with respect to their common global goal. As an efficiency measure for simulating implicate situations involving many events, we develop a noble implementation technique for effectively visualizing parallel (composite) actions and handling abrupt behavioral changes.

Actions are not interruptible, i.e., atomic and their sequential order with respect to their entireties is the only way they are related to each other in most interactive narratives. Coupling between actions is rationalized by parameterizing behavior tree [13], mainly for code reuse in the context of a single event rather than interplay between independent events. While, we have developed a mechanism that allows complex actions to be constructed by reusing primitive motions and enables an agent to promptly react to changes in the ambient states.

The animation technique for a motion to progress to its goal state independently of its current state fulfills the requirements to deal with countless variations of situations in our virtual world. This effective animation technique naturally enables the actions comprising several motions to be simulated with a *similar efficacy*. Considering, for instance, three actions each with three motions, existing animation methods would need to consider 27 cases (even excluding their procedure parts) to author a situation involving three actions. This is compared to 9 cases required in our model. Also, our model achieves the proportional implementation scalability by employing the motions as the atomic units to progressively animate: composite motion, parallel and concurrent actions, and finally events. Furthermore, this uniform use of animation unit enables actions or events to *improvise abrupt changes* in kaleidoscopic virtual world.

The mechanism is formulated as [14]

$$\prod_{i}^{n}(f : g_i \rightarrow m_i)$$

where Π, g and m each denote a temporal sequence, an angle of a joint, and a primitive motion.

An action is performed by iterating a primitive or composite motion or executing a sequence of motions, i.e., the progression of a simulation related to actions is formulated as

$$S(t+\Delta t) = S(t) + \prod_{i}^{M}\left(\sum_{j}^{N} m_j^i, such\ as\ \mathcal{A}_s \supset m_j^i\right)$$

where S, m, $\mathcal{A}$, M, N, $\prod$, and Σ each denote a state, a motion, an action, the number of phases and the number of motions in each phase, a phasic development and a set of parallel motions [14].

We also developed *parallel action* and *concurrent action* by combining actions instead of assembling primitive motions. A parallel action performs two or more individual actions at the same time according to priority. The priority is evaluated between them in the unit of body part, which is to be used exclusively by either one of the individual actions. A concurrent action refers to performing two or more independent actions alternately at a high frequency. While those independent actions cannot be executed simultaneously, they are interleaved to effectively proceed in parallel.

3.2 Identifying events needed to achieve goal Equations

A goal is a situation an agent intends to be in to fulfill her wish or obligation. The goal situation could be one that is newly created or preserved as it is. Unless such a goal is satisfied with the given conditions, some event needs to be performed against the given condition in order to achieve the goal. Such an event in general is complex enough to demand a deliberate planning with smaller events selected by its agent. Those events are successively identified, starting from the macroscopic one that is by itself able to satisfy the goal judged by the functional association such that the effect of an event produces a part of precondition of another event.

A schematic action is repeated or continued to be instantiated into a plan. A schematic action can be formulated as follows, a $\triangleq (S_P, \Delta S, \overline{S_I})\ and\ (\overline{S_I} + \Delta S \rightarrow \overline{S_F})$ where S_P denotes the precondition and ΔS denotes a change in situation, An over bar denotes an average or a typical value below, $\overline{S_I}$ denotes a typical initial situation and $\overline{S_F}$ denotes a typical final situation. All those elements are situations. Likewise, a schematic event can be formulated as follows, $A_i \triangleq \left(S_P^i, \Delta S_i; \overline{S_I^i}\right) and\ \left(\overline{S_I^i} + \Delta S_i \rightarrow \overline{S_F^i}\right)$, That is, an event changes its antecedent situation to its consequent situation. The consequent situation is specified in terms of its procedure and effect parts. Its procedure part describes transitional consequences while its effect part is final consequences.

Once the overall event has been identified in terms of its precondition and effects, the initial overall plan is to be laid out via backward reasoning with respect to its precondition. Specifically, the causality or the premise relation is exploited to deduce the events for a goal in a backward search. This search and selection is recursively applied until all the derived intermediate goals are satisfiable with the (expected) average background conditions. The events identified through the search are converted into actions, which in turn are arranged into the queue for each agent. The actions from the queue are visualized in order for implementing the procedure toward the goal.

4. EXECUTING PLAN

A large complicated event typical of our planning target tends to involve multiple agents with their respective models of virtual world and independent planning capabilities. To execute a plan for an event to achieve a goal, the roles in the plan need to be cast from their associated pools of qualified agents. The moment an agent is cast in a role, that particular agent from the world becomes coupled with the schematic plan. That is, this casting of agents would entail elaboration and augmentation of the original plan according to their individual characteristics and respective schedules preexisting independent of that plan.

Specifically, the plan is to be augmented by adding each cast agent's own and external factors of relevance that were not specified in the original plan but might indirectly affect the original event via the cast agent. Those additional factors are linked to the original event in the plan via their associated agent's relationships with the cast agent. That is, the planning could be compounded by exogenous events, which may have to be added to augment the 'skeletal' plan without contribution to achieving the main goal. The resulting final plan comprises derived events in addition to the main event from the initial goal, forming a graph of events interconnected via their common conditional factors or other types of association between events. That is, $If\ E(A_i \in \prod_i^N A_i) \cap E(A_j \notin \prod_i^N A_i) \neq \emptyset, \prod_i^N + A_j \longrightarrow \prod_i^{N+1} A_i$, where $E(A)$ denotes the agents cast for A.

4.1 Background world

As the common background for all the events and associated agents and conditions, the world in our simulation has greater significance than those used as passive backdrops [16]. That is, narrative worlds are usually simplified in abstract forms or minimized in the forms of spatial configurations, which are geared to serving as stage or environment for particular stories or behaviors in small domains [1], [17], while our full-blown virtual world, a sophisticated version of Working Memory is the central source of user experience as the common background stage for numerous events (or stories) to unfold in. In fact, the background world and events therein are not separated from, but integrated with each other. As a result, events regardless of being main or not and their environments, are likewise parts of the unifying 'active' background world. We pursue *comprehensiveness and sophistication* of the world composition rather than efficiency of its generation in constructing story world [1], [16]. To account for its complex nature the entire world is modeled in terms of entities and their interrelationships in multiple layers, i.e., the reality composed of the physical and social worlds and thereover the conceptual worlds of its inhabitants or agents. World = <R, {Ci}> where R denotes the reality; i=1,2,3,…, # of agents; Agent i's conceptual world Ci = Mi(R), i.e., R as modeled by its modeling function Mi, and Ck∈R for all k≠I (of self), which is described in detail in [9].

4.2 Execution of events

In a technical perspective, our event planning starts with a recursive decomposition process generating a hierarchically structured plan of events sequenced in partial order. Each event that has been identified in this 'horizontal' planning phase is recursively decomposed until all the resulting subsidiary events reduce to the primitive events of actions. Events in our model are integrated as parts of the unifying background world and conversely they collectively form the world. All the relationships including event occurrences are coupled with each other via their preexisting common background world.

Once a main event has been recursively decomposed in terms of actions its leading agent assigns each subsidiary event to a candidate agent from the background world. An agent may be assigned many events if it is judged to be available for them in time context (and in other contexts). That is, $If\ T(A_i) \cap T(\prod_{k=1}^K A_k^H) = \emptyset$, $\prod_{k=1}^K A_k^H + A_i \rightarrow \prod_{k=1}^{K+1} A_k^H$, where $T()$ denotes the time span of, and A_i denotes an arranged part of the main plan, A_k^H denotes the events undertaken by H in the main plan.

The plan fragments constituting a main event are partially ordered according to their relative order with respect to its execution toward the goal. In general plan fragments are decomposed into a tree as illustrated in Fig. 1, and each can start to progress only if its preceding fragments connected through causality have all been successfully completed. That is, $If\ \forall_i \widehat{A_i},\ A \not\succ \widehat{A_i}$, where $\widehat{A_i} \in$ the set of unfinished events. Incomparable events may always be executed in parallel without regard to the others among themselves. $\{\widehat{A_i}\}$ shrinks as the occurrence progresses.

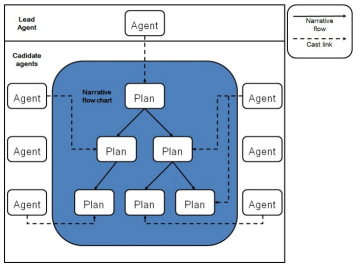

Fig. 1. Example event flow in terms of subsidiary plans and their associated agents

Event execution proceeds in the following steps (greatly detailed in [9]),

(1) Casting event roles.

Before filling the roles of an event their candidate agents' availability needs to be checked in a temporal context. Those candidate agents chosen for the roles in an event plan likely have their respective plans independent of the plan. That is, the agent E can be cast for event A such that $T(A) \subseteq \cap_{i=1}^m \overline{T(A_i)}$, where m denotes the number of events E is involved in.

(2) An event is recursively decomposed until all the resulting subsidiary events reduce to the primitive events of actions. The schematic planning would proceed in several phases such as

i) Find events $\{A_j\}$ such that $\overline{S_F^j} \cap S_P^{i-1} \neq \emptyset$, where i denotes the i-th round of search starting from round *0* for the overall event A_0.

ii) Select the best one $\dot{A_j}$ from $\{A_j\}$.

iii) Collect the events found in (1) for each $s_j \in S_P^i$, to form the candidate event set $\{\dot{A_j}\}$ such that $\overline{S_I^i} + \Delta S_j \rightarrow \overline{S_F^j}$ *and* $\overline{S_F^j} \supseteq S_G^{j-1}$.

iv) Sequence $\{\dot{A_j}\}$ into $\prod_{k=0}^N A_k$, where $N = |\{\dot{A_j}\}|$

The effects resulting from the identified events generally include side effects besides the effects required for the goal. The effects that are not part of the goal S_G are referred to as side effects, i.e., $S_f = S_F - S_G$. Those side effects might be detrimental enough to scuttle the entire plan.

(3) Distribute plan fragment to cast agents such that

$$\{H^1, H^2, H^3 \dots H^m\} = E\left(\prod_j^n \dot{A}_j\right),\ \prod_{k=1}^n A_k^{H^m} + \prod_j^n \dot{A}_j$$

(4) Compare among events in terms of priority.

Select the highest-priority event $\dot{A}_k$ such that $If \dot{A}_k \in A_n^L, \dot{A}_k \succ \prod_{n=1}^{N_T} \hat{A}_{k-n}^L$, where A_n^L denotes an arranged event in the main plan.

4.3 Horizontal and vertical planning

Planning in our model consists of horizontal planning phase and vertical planning phase. The horizontal planning with a goal given successively derives all the events required for satisfying the precondition of each identified event. The lead agent searches its knowledge for candidate agents to assume those identified events. Some qualified agents are given carte blanche for their associated subsidiary events. In vertical planning, each subsidiary event identified in horizontal planning is recursively decomposed until it all reduces to actions to be animated. Each subcontractor agent continually checks the precondition of the subsidiary event it is contracted to. Also, the part of the preconditions related to those actions need to be checked during their respective execution periods.

While a phenomenon (or conditioned event) is unconditionally affected by any relevant condition, an intentional event is affected by some conditions only when they are known by its associated agent. Our model uses two methods to check if event conditions are valid. The first polling-based method checks the conditions at a fixed interval regardless of the planned execution time of the event, and the other method checks the same number of times per event. The efficiency and complexity can be computed as,

$\frac{\left(\frac{|T(A^0)|}{R}\right)}{N}$ and $\frac{\sum_{i=1}^{N} |T(A_i)| \cdot V(S_P^i)}{R}$, for the polling-based method

$\frac{\sum_{i=1}^{k}\left(\frac{|T(A_i)|}{P}\right)}{N}$ and $N \cdot P \cdot \overline{V(S_P^l)}$ for the other method

where A_i denotes subsidiary event, N denotes the number of subsidiary events included in an event, P denotes the number of checks performed per event, R denotes polling interval, V() denotes the number of used variables of.

The polling-based checking is not always appropriate, in that a fixed checking interval could be too frequent for slow-pace events or too infrequent for fast-pace events. The other checking method adjusts the checking interval according to the duration of an event, reflecting the rule of thumb that the shorter duration an event is of the faster the event progresses. This adjustable method generally performs better for intentional events, while the polling method is suitable for fast-changing conditioned events, i.e., natural phenomena. The conditions under which a subsidiary event is interrupted include when no candidate agent is available and when its precondition is not satisfied. An intentional event waits in suspension until its precondition is satisfied by a repair or an alternative. In case it is judged with little chance for the condition to be remedied, the event fails and the failure propagates successively upward to its overarching events.

4.4 Multi-queue based execution of plan

Each event that has been derived in horizontal planning is decomposed in order into actions, which are arranged into its associated agent's queue. This queue could contain different sets of actions, as an agent in general is cast into several events. While the execution order among the events is adjusted according to their priority, the order of actions to be visualized is in fact fixed according to that determined in vertical planning. As a result the outcome expected with the vertical planning is guaranteed regardless of the adjusted order to the priority. Events with lower priority wait in the queue for their order of execution to materialize. Meanwhile those events whose preconditions are not currently satisfied are set to disabled in order not to contend with exogenous events for their chance to execute. Still the continual check on the condition needs to be made for the disabled events as well as other active events. As soon as the conditions are valid those disabled are allowed to resume their contention for priority. In case their disabled time is prolonged beyond their time limit, they are removed from their respective queues. Whereas all the (future) events have been identified through horizontal and vertical planning, their actual deployment in the queues is conducted only after checking their associated preconditions.

In an example situation as depicted in Fig.2, Agent A is partying with its neighbors at home, Agent B is watching for a chance to burglarize A's home, and a policeman happens to be on a patrol around. The situation unfolds as: B attempts to break into A's home but A happens to witness the scene and call the police, leading to B's arrest. If A has been in the protagonist role and is temporarily out of the role, all related subsidiary events would be suspended in line with the original event until the exogenous event is complete. As A reports the theft to the police, a new derived event occurs to arrest the thief with A, B and C cast into its roles. That is, in case an exogenous event involving an agent A is executed it depends on its agent's role in its associated main event whether the main event is suspended or continues to progress. If it is a protagonist role the main event would generally be suspended and otherwise would go on. In case of suspending, $If\ E(A_x) = \emptyset \rightarrow A_x(T) \rightarrow A_x(T + \Delta T_d)\ with\ T2_d - T1_d = \Delta T_d$, where A denotes event, $T1$ denotes start time of an event, $T2$ denotes (planned) end time; x and d indicate main event and exogenous event, respectively. This 'arrest' event is compared on priority in the queue with those existing events C has been involved in, but the 'arrest' event may not actually occur if C has another event with a higher priority he needs to execute. Consequently, the current event of theft is suspended by a derived event of 'escape' regardless of the result of the 'arrest' event. In this case, the 'escape' event derived from the 'arrest' event involving C should be immediately executed on B's realizing the deontic implications of the 'arrest' event. A derived event may entail a succession of derived events due to its consequences. While an agent is idling waiting for its preceding events to be completed it is to be allowed to exploit the idling time to execute, if given, other events. For this, priority contention is limited to among exogenous events.

Scene 1 shows a situation where a derived event E3 is generated by house owner H5 recognizing 'theft' and ‘call police'. If H5 is in an (irreplaceable) protagonist role in E1, the original event 'party' is excluded from priority contention among events by setting disabled, and the original event remains in the queue while checking whether its related preconditions are satisfied again. Therefore, currently-eligible events would be chosen to be processed according to their priority by H3 and H4 cast into the roles of original event 'party'. If H5 is not in a protagonist role, the original event of H3 and H4 can be processed by another agent cast into the role instead of H5 or may continue without the role being cast. If H5 calls 'police', a derived event 'arrest' can be generated by the police on H5 recognizing the call through his recognition module. In this case the causality instance on E3 is due to H1 and H2's event E2 affecting the house ownership of H5. In Scene 2, E3 is terminated by the derived event generated on recognition of E3 by 'policeman' H6 and, accordingly removed and E4 is put in H5 and H6's queues. E4 is an event derived by E3 through a deontic relationship. In case H6 cannot recognize in time H5’s call due to the conditions of H5 and H6, H1 and H2 could go on uninterrupted to successfully finish their event as planned, leading to H5’s loss. Scene 3 shows a process where H1 and H2 execute a derived event E5 on their recognition of H6. Like E3, E5 begins the moment H1 or H2 recognizes H6’s action, and E5 won’t occur if E2 is completed before H6 arrives at the site of E2. Identical event id numbers on different agents’ queues imply linkage in executing suspended events or other planned events. Despite of identical id numbers across queues, the actual execution timing of actions may differ depending on their specific procedures.

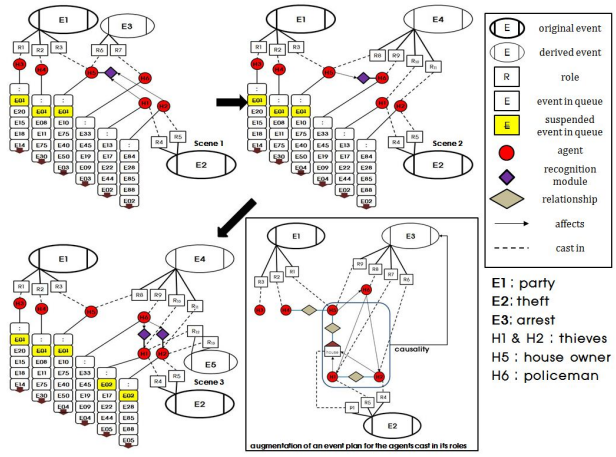

Fig. 2. An Event-level depiction of the procedure to augment an event plan with exogenous events

Fig. 3 describes in the action units ‘party’ event prior to Scene 1 of Fig. 2. The actions in an event are suspended and resumed according to background situations. When an action execution terminates successfully, it is removed from the queue, whereby the suspended event can be resumed from the interrupted action (without need to restart the entire event because events are executed in the unit of agent’s action).

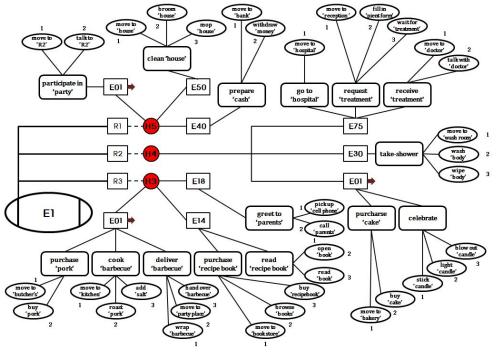

Fig. 3. A Hierarchical composition of an example event

Conditions for event coupling: A coupling between independent events starts with a coincidental contact between them. The contact point is one of the instances comprising the background world, and the contact occurs in the form that the instance affects, or is affected by, those events. Specifically, the first condition for a coupling to happen is for those involved events to overlap on the same instance with respect to their roles. However, this overlap implies only possibility of their coupling, and further elaboration of the (preliminary) overlap along the time axis is needed for an actual coupling. This overlap on an instance is substantialized in diverse forms. For instance, a collision between two cars could be modeled as two location (roles) for two car-running (events) overlapping on the same part of a road (instance), while in case of a robbery, the same goods is modeled to simultaneously play two roles (i.e., targets of robbery and ownership) from independent event and relationship (i.e., a robbery and an ownership.) This overlapping on an instance could be progressively elaborated in finer granularities (e.g., an entire car to its door to its door handle, etc.) with respect to its numerous aspects, eventually leading to modelling at the attribute level (e.g., color, area damaged.)

Spatial colocation: An action terminates when it accomplishes its share of the world’s state corresponding to the goal of its associated event. The changes in the virtual world as a result of actions affect other events through their effects on the background world. The actions related to ‘move’ in particular satisfy part of their execution conditions of a subsidiary event involving ‘move’. In our planning, it is reasonable to model every action is premised on spatial movement (Note a stay is its special case.) The effect of a spatial movement is not sustained on the background world once a subsequent event occurs. In case a subsidiary event is resumed after suspension caused by another event, an action leading to spatial movement unlike other actions is not removed from the queue, if successfully completed, rather it is re-executed to account for the change that has so far in its overarching event happened to the background world. As a result, the timing a movement-causing action is removed from the queue is the moment not the action itself but its associated subsidiary event is completed. While this situation might correspond to re-planning by its associated agent’s behavior in reality, the above scheme produces an equivalent effects, still being much more efficient.

Parallelism in events: The parallelism in the action level is expanded to the event level by evaluating those actions belonging to the highest-priority event within the same queue and those actions that could be assembled into different events. For instance, 'cook' H3 in Fig. 3 on her way to the butcher's after E1 (an event of cook and delivery) she could simultaneously execute the event of 'call her mom' at the third priority. For this parallel execution of two events, the optional body parts [14] are identified with respect to the currently conducted actions, and compared with the essential body parts for the contending actions in an agent's queue. When both body parts match, those two actions are simultaneously conducted in the level of their essential motions.

As illustrated in Fig. 4, agents might attempt to minimize execution time through parallel or concurrent execution of as many events as possible. Since the primitive unit of integrated events is action, parallel and concurrent execution of actions are possible under the following conditions:

$$If P\left(\prod_{i=0}^{N} a_i\right) \cap P\left(\prod_{j=0}^{N} a_j\right) = \emptyset, T\left(\prod_{i=0}^{N} a_i\right) \cap T\left(\prod_{j=0}^{N} a_j\right) \neq \emptyset \rightarrow parallel\ action.$$

$$If P\left(\prod_{i=0}^{N} a_i\right) \cap P\left(\prod_{j=0}^{N} a_j\right) \neq \emptyset, T\left(\prod_{i=0}^{N} a_i\right) \cap T\left(\prod_{j=0}^{N} a_j\right) \neq \emptyset \rightarrow concurrent\ action.$$

where a denotes action belonging to the agent's capability, $P()$ denotes the body parts used in, and $T()$ denotes time span of execution.

Further, parallel and concurrent executions of events are possible under the following conditions:

$$If\ P\left(\prod_{i=0}^{N} \bar{\bar{A}}_i\right) \cap P\left(\prod_{j=0}^{N} A_j\right) = \emptyset, \bar{\bar{A}}_i \succ \hat{A}_i \wedge A_j \succ \hat{A}_j, S_p^{\bar{\bar{A}}_i} = S_p^{A_j} \rightarrow parallel\ event.$$

$$If\ P\left(\prod_{i=0}^{N} \bar{\bar{A}}_i\right) \cap P\left(\prod_{j=0}^{N} A_j\right) \neq \emptyset, \bar{\bar{A}}_i \succ \hat{A}_i \wedge A_j \succ \hat{A}_j, S_p^{\bar{\bar{A}}_i} = S_p^{A_j} \rightarrow concurrent\ event.$$

where $\bar{\bar{A}}$ denotes a set of actions included in an event with the highest priority (or main event), A denotes a set of actions included in an exogenous event. The difference between action and event levels in parallel and concurrent executions is identical to that between action and event. Following their composition of being a sequential set of actions events cannot be juxtaposed for parallelism if those actions to be juxtaposed are not in their turn or their preconditions conflict with each other.

Fig. 4 depicts our example scenario on the time axis in terms of its subsidiary events. As H3 in Scene 1 is not suspended as soon as H5 is suspended, a subsidiary event cannot synchronize with its overarching event due to the time delay in communication of the suspension of the main event to the agent responsible for its subsidiary event. Meanwhile, like the thieves running away on finding the policeman in Scene 3, an event recognized directly through the (thief) agent's perception module causes the induced event (of runaway) to begin almost immediately at the occurrence of the main event (performed by the policeman). An agent may not be able to finish tasks by their associated deadlines. Our simulation model also reflects the real-world practice that people shorten their work times by concurrent or parallel execution. Parallel actions are useful for reducing the total execution time of an event by simultaneously conducting actions from different subsidiary events.

An event is substantiated with an occurrence instantiated from its associated class. The actual execution time and result of an event vary according to its cast agents' conditions. Further parallel actions could affect its total execution time, and potentially what and how subsequent events might occur. A conspicuous merit of our model is the comprehensive ability to uniformly simulate different categories of events: those events under the umbrella of an implicate but all-preplanned event, those exogenous events derived due to the agents cast in the main event's roles, and those events induced or coupled via their (originally-unplanned) common factors of the background world.

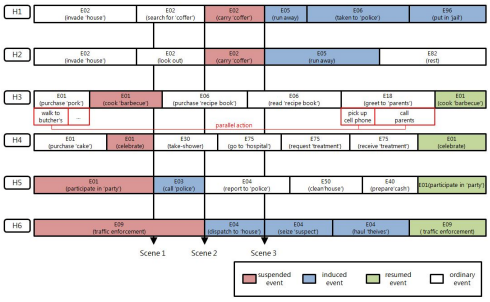

Fig. 4. A timing diagram for example event in execution

Authoring advantages over monolithic authoring: While every event in a conventional IS plan is anticipated in the authoring time (with a pre-contrived indeterminism in a limited scope [10]), our dynamic planning allows unforeseen events to be coincidentally coupled in the execution time. Its key requirement is a set of schematic events (formulated in a reusable form) in the background world. In Fig.5, several schematic events are instantiated with their associated background world factors and accordingly coupled into a progression of a situation. For instance, an obligatory report event occurs due to finding a theft or police chasing event occurs on the thief's escape as a premise for punishment.

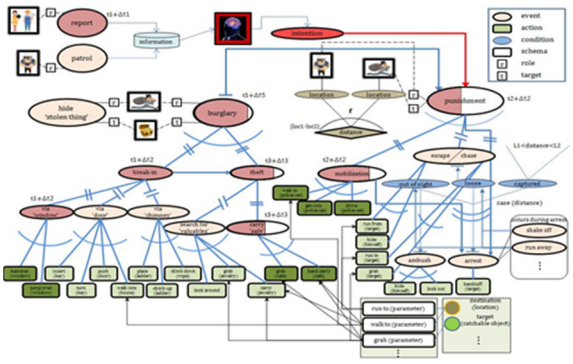

Fig. 5. Authoring burden commensurate to story variability in monolithic authoring

5. IMPLEMENTATION OF EXAMPLE ACTION AND EVENTS

We demonstrate feasibility of our planning method by implementing an example situation comprising independent events, each of which is composed of the animated unit of agents' actions. Each action in turn is constructed by a small set of reusable primitive motions. Consequently, an intricate situation is shown to be implemented in terms of a small set of primitive motions. Specifically, three independent events (i.e., party, theft and arrest) planned and executed by their respective agents come to be intertwined via coupling factors (i.e., ownership, legality, etc.) The visualization of actions like walk and eat, and (a parallel action of) walk while eating, is shown to be efficiently achieved by repetition or sequencing of reusable primitive motions. The efficiency is particularly noticeable when visualizing walk while eating, as a parallel (composite) action between walk and eat. The implementation environment consists of MS Access 2007 Database management system, Open GL graphics tool and Visual Studio 2008.

Fig. 6 shows a sequence of motions to constitute 'walk' toward a goal. If the agent should not be able to arrive at the original destination for some reason, such an abrupt change in progression can be accommodated since our execution model is based on action or motion level. For example, if Agent A with its original plan of moving from C1 to C2 runs into a friend (say, at intermediate location C3 on its way), A would reactively (or autonomously) say hello, and subsequently resume its moving event to C2 according to its original plan. Notice that the original event, 'say hello' event and 'moving from C3 to C2' event are all independent events, which would require to be pre-authored in terms of animated scenes or so in conventional interactive narrative systems.

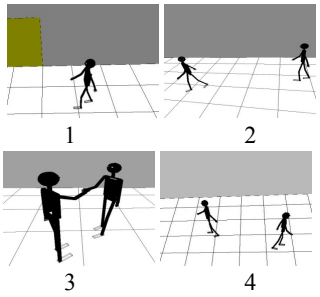

Fig. 6. A sequence of motions for walking and meeting

Fig. 7 shows an agent walking while eating. Parallel or concurrent action is effective since a new composite action is constructed using pre-existing actions (a sequence of motions) rather than using primitive motions. It is applicable to any composite action where each action uses its disjoint set of body parts. For example, eat uses mouth and hand, while walk uses legs. This principle likewise applies to composite events because each event in our planning is decomposed into actions.

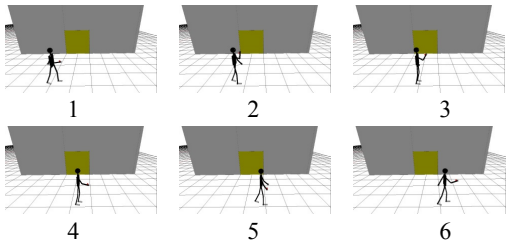

Fig. 7. A sequence of motions for eating while walking

An event shown in Fig. 8 consists of several actions such as phone, talk, walk, grasp, push and climb. A, B and C corresponds to those agents in our earlier example situation involving a party, a burglary and a police arrest. The scenes each show colluding and cooperation between accomplices followed by a police arrest. These scenes are just one of numerous possible sequences, which depend on the conditions of Agents A, B and C and the rest of the background world. Another possible sequence might result from a condition where B is not detected by A, or C was too far away when receiving A's burglary report to get to the scene before B runs away. These independent events are coupled and affect each other, collectively generating diverse situations.

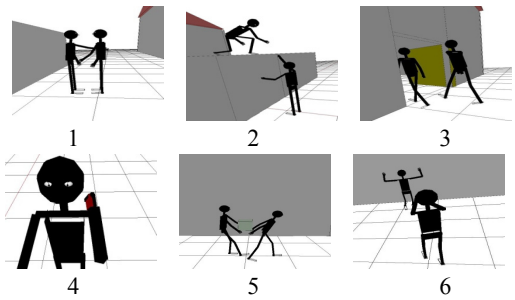

Fig. 8. A sequence of actions for independent events

6. CONCLUSION

We developed an inter-event planning method as a basis for simulating realistic situations to provide abundant pedagogical experience. Our planning aims to generate as diverse realistic situations as possible in an immersed environment in contrast to planning in Interactive Storytelling-based entertainment systems pursuing dramatic interests. Instead of story plot comprising predetermined single-event situations as in conventional IS systems, our inter-event planning method usually involves multiple events coupled via their associated agents' conditions and realistic associations between events in a background world. The specific techniques to realize our method include two-phase planning, autonomous and independent agents, full-blown background world, coupling events via realistic association types, separation of agents from event roles, temporal scheduling, and parallel and concurrent event progression mechanism. Our planning is performed in two directions, i.e., the intra-event phase is based on a vertical decomposition while the inter-event phase is based on horizontal search. The agents are all designed to be an independent (and autonomous) type to behave proactively with

their own belief and planning capability regardless of their roles. As a result a schematic plan is further augmented by conditions associated with those agents cast into the roles of the events identified in the plan. The background world is modelled as a full-blown version to be used as the comprehensive stage for all situations occur in instead of an abstracted or a domain-specific simplified version as a passive backdrop like in most IS-based systems. Events are coupled via meaningful inter-event association types such as deontic associations as well as conventional causality. Coupling patterns are greatly elaborated to further diversify inter-event associations. All those associations are designed as parts of a full-blown background world, which allows exogenous events to be derived and seamlessly (i.e., semantically meaningfully) integrated with the original event. As a result events are separated from the background world (including agents) providing another clue for enhanced diversity of situations. Our model chose agents' actions as the primitive elements of events to allow new events to be efficiently executed in reaction to abrupt changes in relevant conditions. Events in our planning are aligned along time line so as to be scheduled according to their exact timing beyond relative precedence. In line with these strategies we developed a mechanism to determine and inspect in a real-time the execution order and preconditions for events according to varying situation. While events in an overall plan are prioritized or their execution order, the priority is designed to be adjustable to reflect the world conditions in order to simulate the corresponding real-world situations. By designing the action, the animated unit of our implementation, in terms of a sequence of motions, transfer from an action to another action can be unconditionally realized regardless of the initial state or position of its body. As a result actions comprising events coupled via associations are executed independently of results from their preceding actions. Parallel or concurrent actions based on assembling multiple actions beyond multiple motions are extended to simulate parallel and concurrent events without any event being unnecessarily interrupted. We developed an effective execution mechanism for concurrent and parallel actions and expanded for event-level concurrency and parallelism with associated conditions. As a future research, we first need to develop a precise mechanism for coupling events via entities in the sense that each entity itself is modelled to be an active (and complex) concept. Our planning method also suffers, if a lesser degree than conventional narrative systems, from combinatorial explosion of authoring undertaking in constructing its background world.

ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(Ministry of Education) (No. 2016R1D1A2B03935650).

REFERENCES

[1] A. Zook, S. Lee-Urban, M. Rield, H. Holden, R. Sottilare, and K. Brawner, "Automated scenario generation: toward tailored and optimized military training in virtual environments," Proc of the International Conference on the Foundations of Digital Games, ACM, 2012.

[2] L. Amaral and D. Meurers, "From recording linguistic competence to supporting inferences about language acquisition in context," Computer Assisted Language Learning, vol. 21, no. 4, 2008, pp. 323-338.

[3] W. Johnson, L. S. Marsella, and H. Vilhjálmsson, "The DARWARS Tactical Language Training System," Proc of the 26th Interservice/Industry Training, Simulation, and Education Conference, Orlando, FL., 2004.

[4] M. Riedl and R. Young, "The Importance of Narrative as an Affective Instructional Strategy," Design Recommendations for Adaptive ITS : Adaptive Instructional Strategies, vol. 2, Army Science Lab, 2014.

[5] W. Swartout, R. Hill, J. Gratch, W. L. Johnson, C. Kyriakakis, C. Labore, R. Lindheim, S. Marsella, D. Miraglia, B. Moore, J. Morie, J. Rickel, M. Thiebaux, L. Tuch, R. Whitney, and J. Douglas, "Toward the Holodeck: Integrating graphics, sound, character and story," Int'l Conf. on Autonomous Agents, 2001.

[6] M. Cavazza, F. Charles, and S. J. Mead, "Emergent situations in interactive storytelling," Proc. of ACM Symposium on Applied Computing (ACM-SAC), Madrid, Spain, 2002.

[7] Mei Si, Stacy C. Marsella, and David V. Pynadath, "Evaluating Directorial Control in a Character-Centric Interactive Narrative Framework," Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS), Toronto, Canada, 2010, pp. 1289-1296.

[8] R. Hodhod, P. Cairns, and D. Kudenko, "Innovative integrated architecture for educational games: challenges and merits," Transactions on edutainment v. Springer Berlin Heidelberg, 2011, vol. 5, pp. 1-34.

[9] J. Park, "Implementation of an Agent-Centric Planning of Complex Events as Objects of Pedagogical Experiences in Virtual World," Int'l Journal of Contents, vol. 12, no. 1, 2016, pp. 25-43.

[10] F. Charles, M. Lozano, S. J. Mead, A. F. Bisquerra, and M. Cavazza, "Planning formalisms and authoring in interactive storytelling," Proceedings of TIDSE, vol. 3, 2003, pp. 216-225.

[11] K. Erol, *Hierarchical Task Network Planning: Formalization, Analysis, and Implementation*, Dissertation, Univ. of Maryland, 1996.

[12] J. Porteous, M. Cavazza, and F. Charles, "Applying planning to interactive storytelling: Narrative control using state constraints," ACM Trans. on Intelligent Systems and Technology, vol. 1, no.2, Nov. 2010.

[13] A. Shoulson, F. M. Carcia, M. Jones, R. Mead and N. I. Badler., "Parameterizing Behavior Trees," In Motion In Games, Springer, 2011, pp. 144-155.

[14] J. Choi and J. Park, "An Effective implementation of agent's complex actions by reusing primitive motions," Proc. Simultech, Vienna, Austria, 2014.

[15] J. Thomas and M. Young, "Becoming Scientists: Employing Adaptive Interactive Narrative to Guide Discovery Learning," Proc of AIED-07 Workshop on Narrative Learning Environments. Marina Del Rey, CA, USA, 2007.
[16] K. Hartsook, A. Zook, S. Das, and M. Riedl, "Toward supporting stories with procedurally generated game worlds," Proceedings of the 2011 IEEE Conference on Computational Intelligence in Games, 2011.
[17] M. Kapadia, S. Singh, G. Reinman, and P. Faloutsos, "A behavior-authoring framework for multiactor simulations," Computer Graphics and Applications, vol. 31, no. 6, 2011, pp. 44-55.

Jong Hee Park
He acquired his Ph.D in Computer Engineering from Univ. of Florida in 1990. He has since been a professor at Kyungpook Nat'l Univ. in Korea. His research interests focuses on intelligent systems based on cyber-worlds. The specific research topics encompass intelligent tutoring system for immersed language learning, futuristic simulation games, and simulation of cyber-worlds for diverse applications.

Jun Seong Choi
He received his B.S. in Electronics Enginnering from Kyuongpook National University, Korea in 2014, Currently Ph.D course student, in Kyungpook National University, Korea. His research interest is Artificial Intelligence, specifically animation techniques in multi-event