ELSEVIER

Fuzzy Sets and Systems 71 (1995) 155–163

FUZZY sets and systems

Cognitive-map-based decision analysis based on NPN logics☆

Shyi-Ming Chen

Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, ROC

Received November 1993; revised August 1994

Abstract

This paper presents a new method for cognitive-map-based decision analysis based on negative–positive–neutral (NPN) logics, where an efficient algorithm is proposed for performing interval-based inexact reasoning automatically. The algorithm performs inexact reasoning via constructing a sprouting tree. The time complexity of the proposed algorithm is $O(nm)$, where n is the number of cognitive units in the cognitive map, m is the length of a closure path, and $\max(m) = 2n$.

Keywords: Cognitive map; Cognitive unit; Inexact reasoning; NPN logic; Sprouting tree.

1. Introduction

Zhang et al. [15, 18–20] presented the ideas of negative–positive–neutral (NPN) logics and NPN relations. NPN logics and relations assume logic values in the interval $[-1, +1]$ instead of values in $[0, +1]$. Zhang et al. [18] presented a system called Pool2 for cognitive maps [1, 14, 16, 17, 19, 20] development and decision analysis based on NPN logics and NPN relations, where cognitive maps are represented by NPN fuzzy relations. They used a heuristic transitive closure algorithm (HTC) to compute the transitive closure of an NPN fuzzy relation, and a heuristic path searching algorithm (HP) is used to find the positive maximum effect paths and the negative maximum effect paths between any two cognitive units of a cognitive map. The time complexities of the algorithms HTC and HP presented in [18] are $O(n^3)$, respectively, where n is the number of cognitive units in a cognitive map. The system Pool2 is time efficient for applications with small- or medium-sized cognitive maps [18]. However, when the size of the cognitive map is very large, the system Pool2 becomes very inefficient due to the fact that to calculate the heuristic transitive closure of a very large NPN fuzzy relation is a very tedious work. Another drawback of Pool2 is that when the cognitive map is changed, the corresponding NPN fuzzy relation will be changed, and the transitive closure of the NPN fuzzy relation must be recalculated again. According to [18], if $\bar{R}$ is the heuristic transitive closure of a NPN fuzzy relation R and the $[s, j]$th element of $\bar{R}$ is $[f, g]$, then it indicates that the negative and the positive maximum effects of the cognitive unit C_j caused by the cognitive unit C_s are f and g,

☆ This work was supported in part by the National Science Council, Republic of China, under Grant NSC 83-0408-E009-041.

SSDI 0165-0114(94)00265-7

respectively, where $-1 \leqslant f \leqslant g \leqslant +1$. In this case, C_s and C_j are called the starting cognitive unit and the goal cognitive unit, respectively. In [18], the system Pool2 assumed that the truth value of the starting cognitive unit in a cognitive map is absolutely true (i.e., with truth value $+1$). However, if we can allow the truth value of the starting cognitive unit to be an interval $[a, b]$, where $-1 \leqslant a \leqslant b+1$, then there is a room for more flexibility, where -1 represents absolutely false, $+1$ represents absolutely true, 0 represents neutral, the values between 0 and $+1$ represent partially true, and the values between -1 and 0 represent partially false.

In this paper, we propose a new method for cognitive-map-based decision analysis. Our approach allows the truth value of the starting cognitive unit of a cognitive map to be an interval $[a, b]$, where $-1 \leqslant a \leqslant b \leqslant +1$. Furthermore, our approach can overcome the drawbacks of Pool2 mentioned above. We will present an inexact reasoning algorithm to evaluate the positive and the negative maximum effects of the goal cognitive unit C_j caused by the starting cognitive unit C_s in a cognitive map. Furthermore, the positive and the negative maximum effect paths can also be found by the proposed algorithm. The time complexity of the proposed algorithm is $O(nm)$, where n is the number of cognitive units in a cognitive map, m is the length of a closure path, and $\max(m) = 2n$. Our method for cognitive-map-based decision analysis has the advantages of low time complexity and more flexibility.

2. Cognitive maps and NPN logics

Zhang et al. [18] have pointed out that cognitive maps can be used to represent the relationships among the attributes and/or concepts of a given environment, where the relationships are numerically characterized and may be positive, negative, or neutral, where the numerical values associated with the edges between the concepts can be used to indicate the degree of strength with which one concept affects another. A cognitive map (CM) can be defined by 4-tuple,

$$\mathrm{CM} = (C, E, \alpha, \beta),$$

where

(1) C is a finite set of cognitive units (i.e., concepts), $C = \{C_1, C_2, \ldots, C_n\}$;
(2) E is a finite set of directed edges between cognitive units, $E = \{e_1, e_2, \ldots, e_m\}$;
(3) α is a mapping function from cognitive units to an interval $[a, b]$, where $-1 \leqslant a \leqslant b \leqslant +1$;
(4) $\beta: E \to [-1, +1]$ is a mapping function from directed edges to real values between -1 and $+1$.

For example, Fig. 1(a) displays a cognitive map for a public health study and Fig. 1(b) displays a fuzzy version (FCM). Both of them are adapted from [19].

Definition 2.1. Let C_i, C_j, and C_k be any three cognitive units in a cognitive map. If $C_i \xrightarrow{\mu} C_k$, where $\mu \in [-1, +1]$, then C_k is called immediate reachable from C_i. If C_k is immediately reachable from C_i and C_j is immediately reachable from C_k, then C_j is called reachable from C_i. The reachability relationship is the reflexive transitive closure of the immediately reachable relationship.

Definition 2.2. The set of cognitive units which is immediately reachable from a cognitive unit C_i is called the immediate reachability set of C_i and is denoted by $\mathrm{IRS}(C_i)$.

Definition 2.3. The set of cognitive units which is reachable from a cognitive unit C_i is called the reachability set of C_i and is denoted by $\mathrm{RS}(C_i)$.

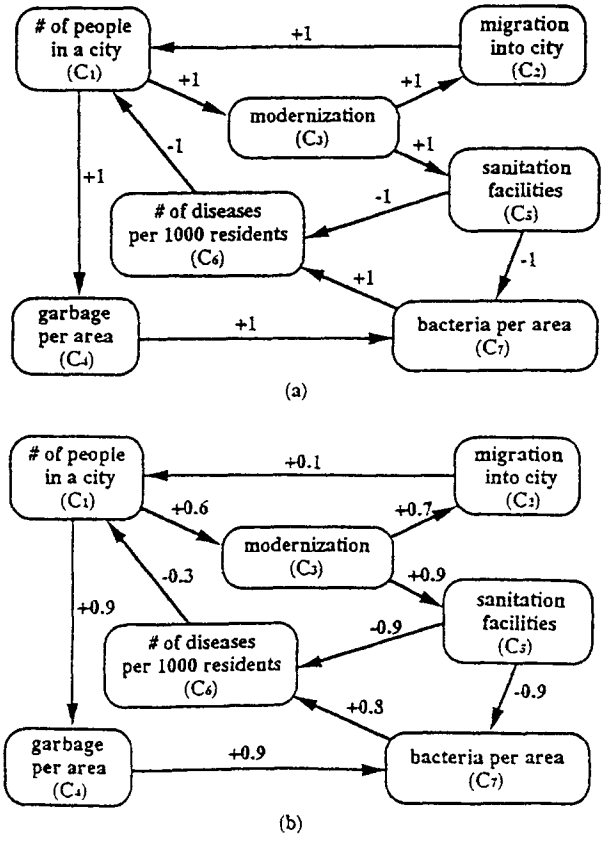

Fig. 1. (a) A crisp cognitive map for public health study. (b) A fuzzy version of the cognitive map in (a).

For example, the immediate reachability set $\mathrm{IRS}(C_5)$ and the reachability set $\mathrm{RS}(C_5)$ for cognitive unit C_5 shown in Fig. 1 are $\{C_6, C_7\}$ and $\{C_1, C_2, C_3, C_4, C_5, C_6, C_7\}$, respectively.

Let V_{ij} denote the value associated with the edge between cognitive units C_i and C_j in a cognitive map. If $V_{ij} = (x, x)$, then it indicates that the value associated with the edge between the cognitive units C_i and C_j is x, where $x \in [-1, +1]$. For example, the value V_{35} associated with the edges between cognitive units C_3 and C_5 shown in Fig. 1(b) is $(+0.9, +0.9)$.

In the following, we briefly review the concepts of NPN logics from [18]. In NPN crisp logic, we have three singleton values -1 (negative), $+1$ (positive), 0 (neutral), and three compound values $(-1, 0)$ (negative or neutral), $(0, +1)$ (neutral or positive), and $(-1, +1)$ (negative or positive or neutral). It is obvious that $(x, x) = x$, where $x \in [-1, +1]$. The compound logical values are pairs ordered (by $\leqslant$) in $\{-1, 0, +1\}$. The truth tables of NPN crisp logic are shown in Fig. 2 which is adapted from [18].

Zhang et al. [18] also presented the ideas of NPN fuzzy logic, where real values between -1 and $+1$ are used for characterizing the strength of a link between two cognitive units. Both NPN crisp logic and NPN fuzzy logic can be described by the following three logic equations:

$$\mathrm{NEG}(x, y) = (\mathrm{NEG}(y), \mathrm{NEG}(x)), \tag{1}$$

$$(x, y)*(u, v) = (\mathrm{Min}(x*u, x*v, y*u, y*v), \mathrm{Max}(x*u, x*v, y*u, y*v)), \tag{2}$$

$$(x, y) + (u, v) = (\mathrm{Min}(x, u), \mathrm{Max}(y, v)), \tag{3}$$

	0	+1	−1	(0, +1)	(−1,0)	(−1, +1)
COM	(−1, +1)	(−1,0)	(0, +1)	−1	+1	0
NEG	0	−1	+1	(−1,0)	(0, +1)	(−1, +1)

(a)

OR	0	+1	−1	(−1,0)	(0, +1)	(−1, +1)
0	0	(0, +1)	(−1,0)	(−1,0)	(0, +1)	(−1, +1)
+1	(0, +1)	+1	(−1, +1)	(−1, +1)	(0, +1)	(−1, +1)
−1	(−1,0)	(−1 +1)	−1	(−1,0)	(−1, +1)	(−1, +1)
(−1,0)	(−1,0)	(−1, +1)	(−1,0)	(−1,0)	(−1, +1)	(−1, +1)
(0, +1)	(0, +1)	(0, +1)	(−1, +1)	(−1, +1)	(0, +1)	(−1, +1)
(−1, +1)	(−1, +1)	(−1, +1)	(−1, +1)	(−1, +1)	(−1, +1)	(−1, +1)

(b)

AND	0	+1	−1	(−1,0)	(0, +1)	(−1, +1)
0	0	0	0	0	0	0
+1	0	+1	−1	(−1,0)	(0, +1)	(−1, +1)
−1	0	−1	+1	(0, +1)	(−1,0)	(−1, +1)
(−1,0)	0	(−1,0)	(0, +1)	(0, +1)	(−1,0)	(−1, +1)
(0, +1)	0	(0, +1)	(−1,0)	(−1,0)	(0, +1)	(−1, +1)
(−1, +1)	0	(−1, +1)	(−1, +1)	(−1, +1)	(−1, +1)	(−1, +1)

(c)

Fig. 2. NPN crisp logic truth table.

where the operator + shown in (3) is the OR operator; the operator $*$ shown in (2) may be any T-norm (such as $\cdot$, $\wedge$ and $\triangle$) extended from the interval $[0, +1]$ to $[-1, +1]$. This extension is made as follows:

$$x*y = \text{Sign}(x)\,\text{Sign}(y)(|x|*|y|), \tag{4}$$

where the function Sign(x) in (4) takes the sign of x, the function Sign(y) takes the sign of y, $x \in [-1, +1]$, and $y \in [-1, +1]$. The T-norms $\cdot$ and $\wedge$ are well known, i.e.,

$$a \cdot b = \text{Sign}(a)\,\text{Sign}(b)(|a|\,|b|),$$

$$a \wedge b = \text{Sign}(a)\,\text{Sign}(b)\,\text{Min}(|a|, |b|),$$

and $\triangle$ is defined as $a \triangle b = \text{Sign}(a)\,\text{Sign}(b)\,\text{Max}(0, |a| + |b| - 1)$, where $a, b \in [-1, +1]$. For example, if $a = 0.8$ and $b = -0.3$, then we can see that

$$a \cdot b = \text{Sign}(0.8)\,\text{Sign}(-0.3)(|0.8|\,|-0.3|) = -0.24,$$

$$a \wedge b = \text{Sign}(0.8)\,\text{Sign}(-0.3)\,\text{Min}(|0.8|, |-0.3|) = -0.3,$$

$$a \triangle b = \text{Sign}(0.8)\,\text{Sign}(-0.3)\,\text{Max}(0, |0.8| + |-0.3| - 1) = -0.1.$$

Table 1
Some identities

Laws	* (T-norm)	+ (OR)
Identity law	$1(x, y) = (x, y)$	Undefined
Null law	$0(x, y) = 0$	$(-1, +1) + (x, y) = (-1, +1)$
Idempotent law	Undefined	$(x, y) + (x, y) = (x, y)$
Commutative law	$(x, y)(u, v) = (u, v)(x, y)$	$(x, y) + (u, v) = (u, v) + (x, y)$
Associative law	$((x, y)(u, v))(w, z) = (x, y)((u, v)(w, z))$	$((x, y) + (u, v)) + (w, z) = (x, y) + ((u, v) + (w, z))$
Distributed law	Undefined	$(x, y)((u, v) + (w, z)) = (x, y)(u, v) + (x, y)(w, z)$

Selection of T-norms is in a domain-dependent manner [18]. Zhang et al. [18] also pointed out that for the NPN logic equations (1)–(3) the identities in Table 1 hold.

3. An algorithm for cognitive-map-based decision analysis

In the following, we propose an interval-based inexact reasoning algorithm for cognitive-map-based decision analysis. Given the interval truth value $[a, b]$ of the starting cognitive unit C_s, the algorithm can perform interval-based inexact reasoning to evaluate the positive and the negative maximum effects of the goal cognitive unit C_j caused by the starting cognitive unit C_s, where $C_s \neq C_j$ and $-1 \leqslant a \leqslant b \leqslant +1$. Furthermore, the positive and the negative maximum effect paths can also be found by the proposed algorithm. The algorithm performs inexact reasoning via constructing a sprouting tree [1, 2]. Each node in the tree is denoted by a triple $(C_k, \mathrm{IRS}(C_k), (y_{k1}, y_{k2}))$, where C_k is a cognitive unit, $\mathrm{IRS}(C_k)$ is the immediate reachability set of C_k, and (y_{k1}, y_{k2}) denotes that the truth value of the cognitive unit C_k is an interval $[y_{k1}, y_{k2}]$, where $-1 \leqslant y_{k1} \leqslant y_{k2} \leqslant +1$. Let V_{ij} denote the value associated with the edge between cognitive units C_i and C_j in a cognitive map. The algorithm is now presented as follows.

Step 1: Select one of the following T-norms for performing inexact reasoning:
(i) $\cdot$
(ii) $\wedge$
(iii) $\triangle$.
Initially, the root node $(C_s, \mathrm{IRS}(C_s), (a, b))$ is a nonterminal node, where C_s is the starting cognitive unit; $\mathrm{IRS}(C_s)$ is the immediate reachability set of C_s; (a, b) indicates that the truth value of the starting cognitive unit C_s entered by the user is an interval $[a, b]$, where $-1 \leqslant a \leqslant b \leqslant +1$.

Step 2: Select one nonterminal node $(C_i, \mathrm{IRS}(C_i), (y_{i1}, y_{i2}))$.
If the goal cognitive unit $C_j \notin \mathrm{RS}(C_i)$, then mark the node as a terminal node.
If the goal cognitive unit $C_j \in \mathrm{IRS}(C_i)$ and $V_{ij} = (\mu, \mu)$, where $\mu \in [-1, +1]$, then create a new node $(C_j, \mathrm{IRS}(C_j), (y_{j1}, y_{j2}))$ in the tree, and an arc, labeled (μ, μ), is directed from the node $(C_i, \mathrm{IRS}(C_i), (y_{i1}, y_{i2}))$ to the node $(C_j, \mathrm{IRS}(C_j), (y_{j1}, y_{j2}))$;
if the selected T-norm is $\cdot$, then
$y_{j1} = \mathrm{Sign}(y_{i1})\,\mathrm{Sign}(\mu)(|y_{i1}|\,|\mu|)$
$y_{j2} = \mathrm{Sign}(y_{i2})\,\mathrm{Sign}(\mu)(|y_{i2}|\,|\mu|)$;
if the selected T-norm is $\wedge$, then
$y_{j1} = \mathrm{Sign}(y_{i1})\,\mathrm{Sign}(\mu)\,\mathrm{Min}(|y_{i1}|, |\mu|)$
$y_{j2} = \mathrm{Sign}(y_{i2})\,\mathrm{Sign}(\mu)\,\mathrm{Min}(|y_{i2}|, |\mu|)$;

if the selected T-norm is Δ, then

$y_{j1} = \text{Sign}(y_{i1})\,\text{Sign}(\mu)\,\text{Max}(0, |y_{i1}| + |\mu| - 1)$

$y_{j2} = \text{Sign}(y_{i2})\,\text{Sign}(\mu)\,\text{Max}(0, |y_{i2}| + |\mu| - 1)$.

If the cognitive unit C_j does not appear on the path from the root node $(C_s, \text{IRS}(C_s), (a, b))$ to the selected node $(C_i, \text{IRS}(C_i), (y_{i1}, y_{i2}))$, then the created node $(C_j, \text{IRS}(C_j), (y_{j1}, y_{j2}))$ is called a success node

else if there is a success node $(C_j, \text{IRS}(C_j), (y_{j1}^*, y_{j2}^*))$ appearing on the path from the root node $(C_s, \text{IRS}(C_s), (a, b))$ to the selected node $(C_i, \text{IRS}(C_i), (y_{i1}, y_{i2}))$, where $-1 \leqslant y_{j1}^* \leqslant y_{j2}^* \leqslant +1$, then the created node $(C_j, \text{IRS}(C_j), (y_{j1}, y_{j2}))$ is called a duplicate node.

If the goal cognitive unit $C_j \notin \text{IRS}(C_i)$ and $C_j \in \text{RS}(C_i)$, then for each cognitive unit $C_k \in \text{IRS}(C_i)$,

if $V_{ik} = (\mu, \mu)$, where $\mu \in [-1, +1]$, then create a new node $(C_k, \text{IRS}(C_k), (y_{k1}, y_{k2}))$ in the tree, and an arc, labeled (μ, μ), is directed from the node $(C_i, \text{IRS}(C_i), (y_{i1}, y_{i2}))$ to the created node $(C_k, \text{IRS}(C_k), (y_{k1}, y_{k2}))$;

if the selected T-norm is $\cdot$, then

$y_{k1} = \text{Sign}(y_{i1})\,\text{Sign}(\mu)(|y_{i1}|\,|\mu|)$

$y_{k2} = \text{Sign}(y_{i2})\,\text{Sign}(\mu)(|y_{i2}|\,|\mu|)$;

if the selected T-norm is $\wedge$, then

$y_{k1} = \text{Sign}(y_{i1})\,\text{Sign}(\mu)\,\text{Min}(|y_{i1}|, |\mu|)$

$y_{k2} = \text{Sign}(y_{i2})\,\text{Sign}(\mu)\,\text{Min}(|y_{i2}|, |\mu|)$;

if the selected T-norm is Δ, then

$y_{k1} = \text{Sign}(y_{i1})\,\text{Sign}(\mu)\,\text{Max}(0, |y_{i1}| + |\mu| - 1)$

$y_{k2} = \text{Sign}(y_{i2})\,\text{Sign}(\mu)\,\text{Max}(0, |y_{i2}| + |\mu| - 1)$.

If there is a node $(C_k, \text{IRS}(C_k), (y_{k1}^*, y_{k2}^*))$ appearing on the path from the root node $(C_s, \text{IRS}(C_s), (a, b))$ to the selected node $(C_i, \text{IRS}(C_i), (y_{i1}, y_{i2}))$, where

$\text{Sign}(y_{k1}) = \text{Sign}(y_{k1}^*)$,

$\text{Sign}(y_{k2}) = \text{Sign}(y_{k2}^*)$,

$|y_{k1}| \leqslant |y_{k1}^*|$, and $|y_{k2}| \leqslant |y_{k2}^*|$,

then the created node $(C_k, \text{IRS}(C_k), (y_{k1}, y_{k2}))$ is called a terminal node. Otherwise, the created node $(C_k, \text{IRS}(C_k), (y_{k1}, y_{k2}))$ is called a nonterminal node.

Step 3: If no nonterminal nodes exist, then go to Step 4. Otherwise, go to Step 2.

Step 4: If there are no success nodes or duplicate nodes, then stop

else let Q be a set formed by success nodes and duplicate nodes.

$Q = \{(C_j, \text{IRS}(C_j), (s_{11}, s_{12})), (C_j, \text{IRS}(C_j), (s_{21}, s_{22})), \ldots, (C_j, \text{IRS}(C_j), (s_{m1}, s_{m2}))\}$

where $s_{ij} \in [-1, +1]$, $1 \leqslant i \leqslant m$, and $1 \leqslant j \leqslant 2$.

Set $f = \text{Min}(s_{11}, s_{21}, \ldots, s_{m1})$.

Set $g = \text{Max}(s_{12}, s_{22}, \ldots, s_{m2})$.

The negative and the positive maximum effects of the goal cognitive unit C_j caused by the starting cognitive unit C_s are f and g, respectively, where $-1 \leqslant f \leqslant g \leqslant +1$. In this case, the node $(C_j, \text{IRS}(C_j), (f, -))$ in the tree is marked as the negative maximum effect node, and the node $(C_j, \text{IRS}(C_j), (-, g))$ in the tree is marked as the positive maximum effect node, where the symbol – denotes "do not care". The ordered sequence of the cognitive units appearing on the path from the root node $(C_s, \text{IRS}(C_s), (a, b))$ to the positive maximum effect node $(C_j, \text{IRS}(C_j), (-, g))$ forms a positive maximum effect path; the ordered sequence of the cognitive units appearing on the path from the root node $(C_s, \text{IRS}(C_s), (a, b))$ to the negative maximum effect node $(C_j, \text{IRS}(C_j), (f, -))$ forms a negative maximum effect path. Display each positive maximum effect path and each negative maximum effect path.

In the following, we use an example to illustrate the inexact reasoning process.

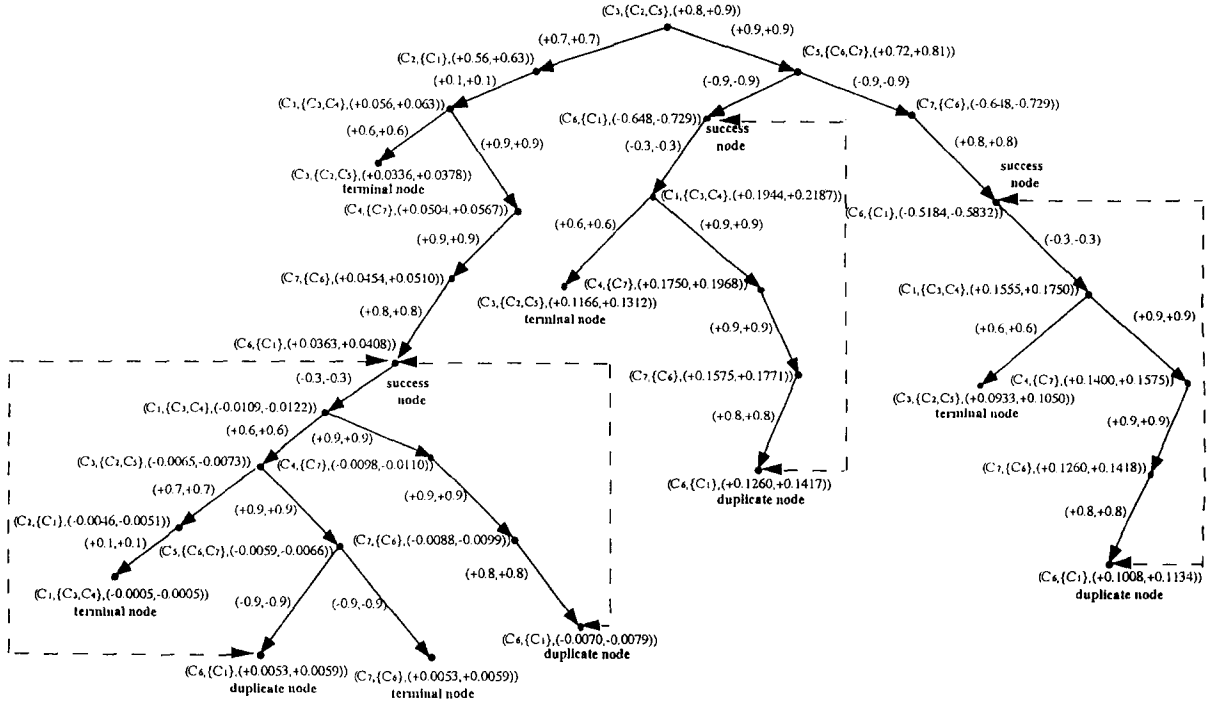

Fig. 3. Sprouting tree of Example 3.1.

Example 3.1. Assume that the truth value of the starting cognitive unit C_3 of the cognitive map shown in Fig. 1(b) is an interval $[+0.8, +0.9]$, and the user wants to know the positive and the negative maximum effects of the goal cognitive unit C_6 caused by the starting cognitive unit C_3. Further assume that the T-norm selected by the user is $\cdot$, then by applying the inexact reasoning algorithm, the tree sprouts as shown in Fig. 3.

From Fig. 3, we can see that there are three success nodes and four duplicate nodes in the tree. Thus, we can obtain the following results:

$$Q = \{(C_6, \{C_1\}, (+0.0363, +0.0408)), (C_6, \{C_1\}, (-0.648, -0.729)),$$
$$(C_6, \{C_1\}, (-0.5184, -0.5832)), (C_6, \{C_1\}, (+0.0053, +0.0059)),$$
$$(C_6, \{C_1\}, (-0.0070, -0.0079)), (C_6, \{C_1\}, (+0.126, +0.1417)),$$
$$(C_6, \{C_1\}, (-0.1008, -0.1134))\},$$

$$f = \text{Min}(+0.0363, -0.648, -0.5184, +0.0053, -0.0070, +0.126, -0.1008)$$
$$= -0.648,$$

$$g = \text{Max}(+0.0408, -0.729, -0.5832, +0.0059, -0.0079, +0.1417, -0.1134)$$
$$= +0.1417.$$

Thus, we can see that the positive and the negative maximum effects of the goal cognitive unit C_6 caused by the starting cognitive unit C_3 are $+0.1417$ and -0.648, respectively. In this case, the node $(C_6, \{C_1\},$

$(+0.126, +0.1417))$ is marked as the positive maximum effect node; the node $(C_6, \{C_1\}, (-0.648, -0.729))$ is marked as the negative maximum effect node. Therefore, the positive maximum effect path can be obtained as follows:

$$C_3 \rightarrow C_5 \rightarrow C_6 \rightarrow C_1 \rightarrow C_4 \rightarrow C_7 \rightarrow C_6.$$

The negative maximum effect path can be obtained as follows:

$$C_3 \rightarrow C_5 \rightarrow C_6.$$

From the above results, we know that the negative effect is much larger than the positive one, so a decision can be reached.

4. Analysis of finiteness problems

A very important property of the proposed algorithm to construct a sprouting tree is that it is finite. The proof of this property requires the following lemma.

Lemma 4.1. *In any infinite directed tree in which each node has only a finite number of direct successors, there is an infinite path leading from the root.*

Proof. See [10, p. 97]. □

Theorem 4.1. *The sprouting tree constructed by the proposed algorithm is finite.*

Proof. The proof is by contradiction. Assume that there exists an infinite sprouting tree. Because the number of direct successors for each node in the tree is limited by $n-1$, were n is the number of cognitive units in a cognitive map, by Lemma 4.1 there is an infinite path $(C_s, \text{IRS}(C_s), (a, b))$, $(C_1, \text{IRS}(C_1), (y_{11}, y_{12}))$, $(C_2, \text{IRS}(C_2), (y_{21}, y_{22}))$, ... from the root node $(C_s, \text{IRS}(C_s), (a, b))$, where (a, b) denotes that the truth value of the starting cognitive unit C_s entered by the user is an interval $[a, b]$, $-1 \leqslant a \leqslant b \leqslant +1$. But in a sprouting tree, the path length from the root node $(C_s, \text{IRS}(C_s), (a, b))$ to either success node or duplicate node is limited by $2n$ based on [18], where n is the number of cognitive units in a cognitive map. This is a contradiction. Proving that an infinite sprouting tree existed was incorrect. □

Theorem 4.2. *The time complexity of the proposed algorithm is* $O(nm)$, *where n is the number of cognitive units, m is the length of a closure path, and* $\max(m) = 2n$.

Proof. Let m denote the length of a closure path. Based on [18], we know that the maximum length of a single branch of a sprouting tree is $2n$ (i.e., $\max(m) = 2n$), where n is the number of cognitive units; given a pair of starting cognitive unit and goal cognitive unit, the algorithm constructs a sprouting with many branches, and the number of direct successors for each node in the tree is limited by $n-1$. Thus, we know that the time complexity of the proposed algorithm is $O(nm)$. □

5. Conclusions

We have presented a new method for cognitive-map-based decision analysis based on NPN logics. An efficient algorithm has been introduced for performing inexact reasoning automatically, where the truth

values of the cognitive units in a cognitive map are represented by an interval $[a, b]$, where $-1 \leqslant a \leqslant b \leqslant +1$. The proposed interval-based inexact reasoning scheme can provide an useful way for cognitive-map-based decision analysis. The time complexity of the proposed algorithm is $O(nm)$, where n is the number of cognitive units in a cognitive map, m is the length of a closure path, and $\max(m) = 2n$. We have implemented a generic decision analysis system [18] based on the proposed algorithm on a PC/AT by using Turbo Pascal version 5.5. Our method is more flexible than the one presented in [18] due to the facts that it allows the truth value of the starting cognitive unit to be an interval $[a, b]$ rather than a crisp value $+1$, where $-1 \leqslant a \leqslant b \leqslant +1$, and it does not need to recalculate the transitive closure of the NPN fuzzy relation when the cognitive map is changed. A direct consequence of our method for cognitive-map-based decision analysis is low time complexity and more flexibility.

References

[1] C.L. Chang, *Introduction to Artificial Intelligence Techniques* (JMA Press, Texas, 1985).

[2] S.M. Chen, A new approach to handling fuzzy decisionmaking problems, *IEEE Trans. System Man Cybernet.* **18** (1988) 1012–1016.

[3] S.M. Chen, An improved algorithm for inexact reasoning based on extended fuzzy production rules, *Cybernet. Systems Internat. J.* **23** (1992) 463–482.

[4] S.M. Chen, J.S. Ke and J.F. Chang, Knowledge representation using fuzzy Petri nets, *IEEE Trans. Knowledge Data Eng.* **2** (1990) 311–319.

[5] S.M. Chen, J.S. Ke and J.F. Chang, An inexact reasoning algorithm for dealing with inexact knowledge, *Internat. J. Software Eng. Knowledge Eng.* **1** (1991) 227–244.

[6] S.M. Chen, J.S. Ke and J.F. Chang, Fuzzy reasoning based on fuzzy Petri nets, *Internat. J. Inform. Management Sci.* **3** (1992) 39–52.

[7] S.M. Chen, A cognitive-map-based approach for decision analysis, in: *Proc. 1993 National Computer Symp.*, Chiayi, Taiwan, ROC (1993) 186–193.

[8] A. Kandel and L. Yelowitz, Fuzzy chains, *IEEE Trans. System Man Cybernet.* **4** (1974) 472–475.

[9] C.G. Looney, Fuzzy Petri nets for rule-based decision-making, *IEEE Trans. System. Man Cybernet.* **18** (1988) 178–183.

[10] J.L. Peterson, *Petri Nets, Theory, and the Modeling of Systems* (Prentice-Hall, Englewood Cliffs, NJ, 1981).

[11] L.A. Zadeh, Fuzzy sets, *Inform. and Control* **8** (1965) 338–353.

[12] L. Zadeh, Fuzzy logic, *IEEE Comput.* **21** (1988) 83–93.

[13] W.R. Zhang, Pool – a semantic model for approximate reasoning and its application in decision support, *J. Management Inform. System* (Special Issue), Decision support and knowledge-based systems **3** (1987) 65–78.

[14] W.R. Zhang, Cognitive map composition, derivation, and focus generation for distributed group decision support, *Proc. 23rd Hawaii Internat. Conf. Syst. Sci.*, Hawaii (1990) 318–326.

[15] W.R. Zhang, NPN calculi: a family of three strict Q-algebras, *Proc. 21st IEEE Internat. Symp. MVL*, Victoria, Canada (1991) 255–261.

[16] W.R. Zhang, J.C. Bezdek and R.O. Pettus, NPN relations and cognitive map development, *Proc. Internat. Symp. on Methodologies for Intelligent Systems*, Charlotte, North Carolina (1987) 271–281.

[17] W.R. Zhang and S.S. Chen, An architecture for cognitive maps, *Proc. 2nd IEEE Int. Conf. on Neural Networks*, San Diego, CA (1988) 231–238.

[18] W.R. Zhang, S.S. Chen and J.C. Bezdek, Pool2 – a generic system for cognitive map development and decision analysis, *IEEE Trans. System Man Cybernet.* **19** (1989) 31–39.

[19] W.R. Zhang, S.S. Chen, K.H. Chen, M. Zhang and J.C. Bezdek, On NPN logic, *Proc. 18th IEEE Internat. Symp. MVL*, Palma de Mallorca, Spain (1988) 381–388.

[20] W.R. Zhang, S.S. Chen, W. Wang and R.S. King, A cognitive-map-based approach to the coordination of distributed cooperative agents, *IEEE Trans. System Man Cybernet* **22** (1992) 103–113.