2012 IEEE International Conference on Systems, Man, and Cybernetics
October 14-17, 2012, COEX, Seoul, Korea

Operations of Grid General Type-2 Fuzzy Sets Based on GPU Computing Platform

Long Thanh Ngo
Department of Information Systems
Le Quy Don Technical University, Hanoi, Vietnam
Email: ngotlong@gmail.com

***Abstract*—Nowadays, many studies have focused on representation of type-2 fuzzy sets for reducing complexity of computation of operations, especially type-2 fuzzy logic systems. Advances of nVIDIA computing technology allow to deploy of general purpose applications of GPU by parallelizing algorithms. This paper deals with an approach to representation of type-2 fuzzy sets by dividing domain into grid, called grid-based T2FS. Grid-based T2FS is easily to understand and deploy on platform of Graphics Processor Units (GPU) computing and Compute Unified Device Architecture(CUDA) by taking the advantages of computation on matrices. Experiments are implemented in various applications involving join, meet, negation operation and defuzzification. Results on runtime and accuracy are summarised in comparison with CPU computation to show efficiency of the approach.**

***Index Terms*—Type-2 fuzzy sets, GPU computing, defuzzification, fuzzy operations.**

I. Introduction

Nowadays, many studies have focused on representation of type-2 fuzzy sets for reducing complexity of computation of operations, especially for type-2 fuzzy logic systems. Because of advantages of type-2 fuzzy sets in management of uncertainty, applications of type-2 fuzzy sets are deployed in many fields. Almost applications are limited by computational complexity of general type-2 fuzzy sets. Recently, these are many researches arising from reduction the complexity of these systems that depend on representation of type-2 fuzzy sets. There are many approaches to representation of type-2 fuzzy sets. The point-based representation [4], [5], [6] has proposed with operations and computation of type-2 fuzzy sets and systems. The point-based pure representation of type-2 fuzzy sets are hardly to apply because of huge cloud of points by discretizing the domain. Mendel et al [6], [7] proposed the representation based on embedded fuzzy sets, vertical slices and gains successful from approximate computation of embedded type-2 fuzzy sets for type-2 fuzzy logic systems. Starczewski [13] introduced a method for complexity reduction of operations on triangular type-2 fuzzy sets. For this purpose, Coupland et al [2] proposed geometric method for representation type-1 and interval type-2 fuzzy sets, new algorithms for various operations on type-1 and type-2 fuzzy sets and for defuzzification. Coupland et al [3] presented new techniques using upper and lower surfaces for performing logical operations on type-2 fuzzy sets with considering computational speed and accuracy. Techniques based on cloud of triangles are limited by computational capability in case of complexity type-2 fuzzy sets with huge quantity of triangles. In [10], [11], [12], approaches to representation of general type-2 fuzzy sets are proposed based on TIN-based geometric representation with or without using upper and lower surfaces, RCTIN-based representation for type-2 fuzzy logic systems. Hagras et al [14], [15], [16] proposed an approach to representation of type-2 fuzzy sets using zSlices that be able to gap interval type-2 fuzzy sets and general type-2 fuzzy sets. A similar approach to representation of type-2 fuzzy sets was proposed by Mendel et al [8] based on α - plane representation.

CUDA-based GPU computing has developed by nVIDIA and gains advantages with capability to parallel algorithms into sub-algorithms on threads, blocks as matrices. General purpose GPU has applied for intelligent computation such as fuzzy c-mean clustering [17], fuzzy neural networks [21], fuzzy ART clustering [20] or fuzzy logic systems [18], [19].

On the basis of matrix-based parallelisation, grid type-2 fuzzy sets are proposed to take this advantage. The paper introduces an approach to approximate representation of general type-2 fuzzy sets, called grid type-2 fuzzy sets, that use a grid to store data at vertices as matrices and value of non-vertex points is interpolated based on four vertices of rectangle containing the point. This representation is the basis to design algorithms of general type-2 fuzzy sets on GPU paralleling platform. The representation also allows to easily deploy algorithms of grid type-2 fuzzy sets involving join, meet, negation and defuzzification operations on GPU. Experiments are implemented on GPU and CPU platforms with summarised reports on various input parameters to show advantages of this approach. Runtime of algorithms on GPU is enough fast to deploy applications of type-2 fuzzy sets to real problems.

The paper is organized as follows: II presents an overview on type-2 fuzzy sets and GPU computing; III introduces grid type-2 fuzzy sets, operations involving join, meet, negation and defuzzification; IV presents experiments on representation and operations with various parameters of input grid type-2 fuzzy sets; V is conclusion and future works.

II. Type-2 Fuzzy Sets and GPU Computing

A. Type-2 Fuzzy Sets

A type-2 fuzzy set in X is denoted $\tilde{A}$, and its membership grade of $x \in X$ is $\mu_{\tilde{A}}(x,u), u \in J_x \subseteq [0,1]$, which is a type-1 fuzzy set in [0, 1]. The elements of domain of $\mu_{\tilde{A}}(x,u)$

978-1-4673-1714-6/12/$31.00 ©2012 IEEE

are called primary memberships of x in $\tilde{A}$ and memberships of primary memberships in $\mu_{\tilde{A}}(x,u)$ are called secondary memberships of x in $\tilde{A}$.

Definition 2.1: A $type-2\ fuzzy\ set$, denoted $\tilde{A}$, is characterized by a type-2 membership function $\mu_{\tilde{A}}(x,u)$ where $x \in X$ and $u \in J_x \subseteq [0,1]$, i.e.,

$$\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u)) | \forall x \in X, \forall u \in J_x \subseteq [0,1]\} \quad (1)$$

or

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u))/(x,u), J_x \subseteq [0,1] \quad (2)$$

in which $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$.

At each value of x, say $x = x'$, the 2-D plane whose axes are u and $\mu_{\tilde{A}}(x',u)$ is called a *vertical slice* of $\mu_{\tilde{A}}(x,u)$. A *secondary membership function* is a vertical slice of $\mu_{\tilde{A}}(x,u)$. It is $\mu_{\tilde{A}}(x = x',u)$ for $x \in X$ and $\forall u \in J_{x'} \subseteq [0,1]$, i.e.

$$\mu_{\tilde{A}}(x = x', u) \equiv \mu_{\tilde{A}}(x') = \int_{u \in J_{x'}} f_{x'}(u)/u, J_{x'} \subseteq [0,1] \quad (3)$$

in which $0 \leq f_{x'}(u) \leq 1$.

Uncertainty of $\tilde{A}$, denoted FOU, is union of primary functions i.e. $FOU(\tilde{A}) = \bigcup_{x \in X} J_x$. Upper/lower bounds of membership function (UMF/LMF), denoted $\overline{\mu}_{\tilde{A}}(x)$ and $\underline{\mu}_{\tilde{A}}(x)$, of $\tilde{A}$ are two type-1 membership function and bounds of FOU.

Let $\tilde{A}, \tilde{B}$ be type-2 fuzzy sets whose secondary membership grades are $f_x(u), g_x(w)$, respectively. Theoretic operations of type-2 fuzzy sets such as union, intersection and complement are described [5] as follows:

$$\mu_{\tilde{A} \cup \tilde{B}}(x) = \mu_{\tilde{A}}(x) \sqcup \mu_{\tilde{B}}(x) = \int_u \int_v (f_x(u) \star g_x(w))/(u \vee w) \quad (4)$$

$$\mu_{\tilde{A} \cap \tilde{B}}(x) = \mu_{\tilde{A}}(x) \sqcap \mu_{\tilde{B}}(x) = \int_u \int_v (f_x(u) \star g_x(w))/(u \star w) \quad (5)$$

$$\mu_{\tilde{A}}(x) = \mu_{\neg \tilde{A}}(x) = \int_u (f_x(u))/(1-u) \quad (6)$$

where $\vee$, $\star$ are t-cornorm, t-norm, respectively. Type-2 fuzzy sets are called an interval type-2 fuzzy sets if the secondary membership function $f_{x'}(u) = 1\ \forall u \in J_x$.

B. GPU Computing

Recently, the GPU has been transformed into the general purpose GPU. The programmable GPU has evolved into a high parallel, multi-threaded, multi-core processor with huge computational power and memory bandwidth. Fig. 1 shows CUDA processing model. CUDA allows multiple kernels to be run simultaneously on a single GPU. CUDA refers to each kernel as a grid. A grid is a collection of blocks. Each block runs the same kernel but is independent of each other. A block contains threads, which are the smallest divisible unit on a GPU.

Fig. 2 shows the CUDA programming model. The model is a set of massive threads that run in parallel. A thread block is a number of SIMD(Single Instruction, Multiple Data) threads that work on an Streaming Multiprocessor at a given time, can exchange information through the shared memory, and can be synchronized. The operations are systematized as a grid of thread blocks. For operation parallelism, the programming model allows a developer to partition a program into several sub-problems, each of which is executed independently on a block. Each sub-program can be further divided into finer pieces that perform the same function for execution on different threads within the block. For dataset parallelism, datasets can be divided into smaller chunks that are stored in the shared memory, and each chunk is visible to all threads of the same block. This local data arrangement approach reduces the need to access off-chip global memory, which reduces data access time.

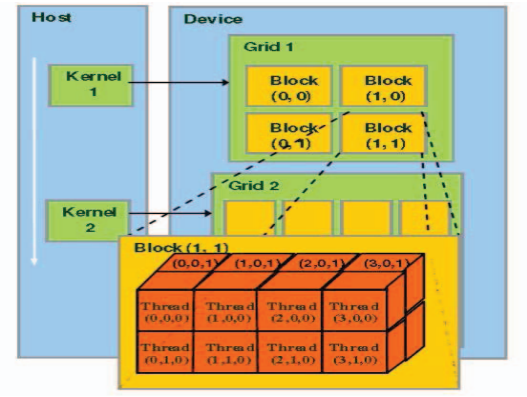

Fig. 1. CUDA processing model design [22]

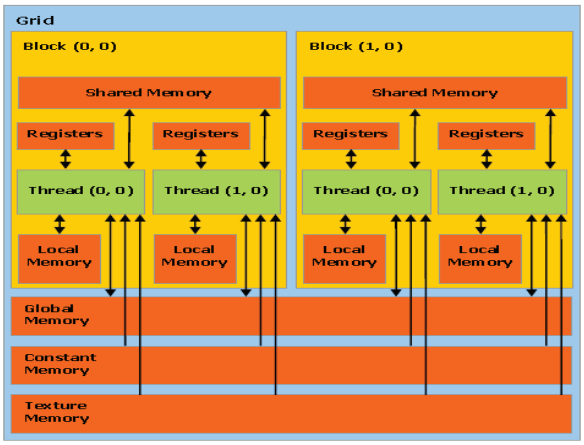

Fig. 2. CUDA GPU memory model design [22]

III. Operations of Grid Type-2 Fuzzy Sets

A. Grid Type-2 Fuzzy Sets

GPU-based computation is to speed up problems that possess the huge complexity of computation. The advantages of this approach is the use matrix - based computation. For this purpose, grid type-2 fuzzy sets (G-T2FS) is developed based on discretizing the domain into the regular grid. Before giving definition of G-T2FS, some concepts are described as follows:

Let X be domain of type-2 fuzzy set $\tilde{A}$ and $U = (u_{min}, u_{max}) \subseteq [0,1]$ be secondary domain of $\tilde{A}$. Suppose that $X = [x_{min}, x_{max}]$. Call dx, du are step of x, u axis, respectively. Space $X \times U$ of $\tilde{A}$ can be divided into a grid that is union of $M \times N$ cells, in which $M = [(x_{max} - x_{min})/dx]$ and $N = [(u_{max} - u_{min})/dx]$. According to this way, a sub type-2 fuzzy set $\tilde{A}_{ij}$ in domain of the cell (i,j) is described as follows:

$$\tilde{A}_{ij} = \{((x,u), \mu_{\tilde{A}}(x,u) | x \in X_i = [\underline{x}_i, \overline{x}_i], u \in U_j = [\underline{u}_j, \overline{u}_j]\} \quad (7)$$

in which $\underline{x}_i = x_{min} + i * dx$, $\overline{x}_i = \underline{x}_i + dx$, $\underline{u}_j = u_{min} + j * du$, $\overline{u}_j = \underline{u}_j + du$, $i = 0, 1, ..., N-1$ and $j = 0, 1, ..., M-1$.

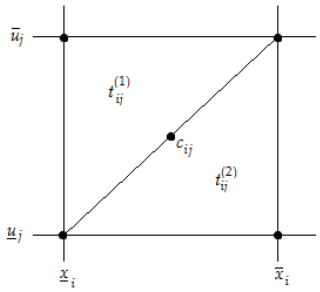

Fig. 3. Definition of the cell (i,j) in grid $X \times U$ space.

To define grid type-2 fuzzy set, called $\tilde{A}^g$, an approximate representation of sub type-2 fuzzy set $\tilde{A}_{ij}$ by a cell type-2 fuzzy set $\tilde{A}_{ij}^c$ is introduced. Let $f_{ij}(x,u)$ be membership grade of $\tilde{A}_{ij}^c$ at (x,u) and described as follows:

Provide that (x,u) is in the cell (i,j) and $d_k (k = \overline{1,4})$ are the distance from (x,u) to k^{th} vertex of the cell, i.e.

$$d_1 = \sqrt{(x - \underline{x})^2 + (u - \underline{u})^2} \quad d_2 = \sqrt{(x - \underline{x})^2 + (u - \overline{u})^2} \quad (8)$$
$$d_3 = \sqrt{(x - \overline{x})^2 + (u - \overline{u})^2} \quad d_4 = \sqrt{(x - \overline{x})^2 + (u - \underline{u})^2}$$

Let u_k be the rate between (x,u) and k^{th} vertex, i.e

$$u_k = \frac{1}{\sum_{i=1}^{4} \sqrt{d_k / d_i}} \quad (9)$$

And the membership grade at (x,u) of $\tilde{A}$ is computed:

$$f_{ij}(x,u) = \sum_{i=1}^{4} u_i \times f^{(i)} / \sum_{i=1}^{4} u_i \quad (10)$$

in which $f^{(1)} = \mu_{\tilde{A}_{ij}}(\underline{x}, \underline{u})$, $f^{(2)} = \mu_{\tilde{A}_{ij}}(\underline{x}, \overline{u})$, $f^{(3)} = \mu_{\tilde{A}_{ij}}(\overline{x},$ $f^{(4)} = \mu_{\tilde{A}_{ij}}(\overline{x}, \underline{u})$.

Definition 3.1: A cell type-2 fuzzy set, denoted $\tilde{A}_{ij}^c$, is approximate representation of sub type-2 fuzzy set $\tilde{A}_{ij}$ and is defined as follows:

$$\tilde{A}_{ij}^c = \{((x,u), \mu_{\tilde{A}_{ij}^c}(x,u)) | x \in [\underline{x}_i, \overline{x}_i], u \in [\underline{u}_i, \overline{u}_i]\} \quad (11)$$

in which $\mu_{\tilde{A}_{ij}^c}(x,u)) = f_{ij}(x,u)$ if $(x,u) \in D_{ij}$ and D_{ij} is domain of the cell (i,j), $i = 0, 1, ..., N-1$, $j = 0, 1, ..., M-1$.

Definition 3.2: A grid type-2 fuzzy set, denoted $\tilde{A}^g$, is union of above defined cell type-2 fuzzy set, i.e

$$\tilde{A}^g = \bigcup_{i=0}^{N-1} \bigcup_{j=0}^{M-1} \tilde{A}_{ij}^c \quad (12)$$

Let a value point $(x,u) \in X \times U$, the membership grade $\mu_{\tilde{A}^g}(x,u)$ is computed as follows:

$$\mu_{\tilde{A}^g}(x,u) = f_{ij}(x,u) \quad (13)$$

in which $(x,u) \in D_{ij}$, $i = [(x - x_{min})/dx]$ and $j = [(u - u_{min})/du]$.

Hence, $\tilde{A}^g$ is representation of $\tilde{A}$ with a degree of approximation. See that high resolution of the grid results in the degree of approximation is small and vice versa. We define the degree of approximation to measure this criteria as follows:

Definition 3.3: The degree of approximation (DoA) is the difference between original type-2 fuzzy set $\tilde{A}$ and the representing grid type-2 fuzzy set $\tilde{A}^g$ and is defined as follows:

$$DoA = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \triangle_{ij} / n_c \quad (14)$$

in which $\triangle_{ij} = |\mu_{\tilde{A}}(x_i^c, u_j^c) - f_{ij}(x_i^c, u_j^c)|$, $x_i^c = \frac{1}{2}(\underline{x}_i + \overline{x}_i)$, $u_i^c = \frac{1}{2}(\underline{u}_i + \overline{u}_i)$ and n_c is number of cells that belong to the FOU of $\tilde{A}$.

A grid type-2 fuzzy set $\tilde{A}^g$ is evaluated by three parameters involving DoA, dx and du. This approach to representation is to support computation based on GPU that allows to parallel matrix of the grid.

Example 3.1: Let $\tilde{A}$ is a general type-2 fuzzy set. The feature membership functions of $\tilde{A}$ are described as follows:

FOU is Gaussian function with upper MF and lower MF as follows:

Upper MF of FOU:

$$f_u(x) = \begin{cases} e^{-\frac{1}{2}(\frac{x-m_1}{\sigma})^2} & if\ x < m_1 \\ 1 & if\ m_1 \le x \le m_2 \\ e^{-\frac{1}{2}(\frac{x-m_2}{\sigma})^2} & if\ x > m_2 \end{cases} \quad (15)$$

Lower MF of FOU:

$$f_l(x) = \begin{cases} e^{-\frac{1}{2}(\frac{x-m_2}{\sigma})^2} & if\ x < \frac{m_1+m_2}{2} \\ e^{-\frac{1}{2}(\frac{x-m_1}{\sigma})^2} & if\ otherwise \end{cases} \quad (16)$$

where $m_1 = 3.0$, $m_2 = 4.0$ and $\sigma = 0.5$.

The next feature of $\tilde{A}$ is set of points where $\mu_{\tilde{A}}(x,u) = 1.0$, involves points belong to the MF described as follows:

$$f_m(x) = e^{-\frac{1}{2}(\frac{x-(m_1+m_2)/2}{\sigma})^2} \quad (17)$$

The secondary membership function at x of $\tilde{A}$ are Gaussian functions that are described as follows:

$$g(u) = \begin{cases} e^{-\frac{1}{2}(\frac{u}{\sigma_1})^2} & if\ u \ge u_0 \\ e^{-\frac{1}{2}(\frac{u}{\sigma_2})^2} & if\ u < u_0 \end{cases} \quad (18)$$

in which $u_0 = f_m(x)$, $\sigma_1 = 3.035 * |\overline{u} - u_0|$ and $\sigma_2 = 3.035 * |\underline{u} - u_0|$.

Fig. 4 depicts a grid type-2 fuzzy set that is made with parameters $X = [1.48, 5.52]$, $U = [0, 1]$, $M = 255(dx = 0.015822956)$, $N = 63(du = 0.015873017)$. $\tilde{A}^g$ is approximate representation of $\tilde{A}$ with the grid involving $M \times N$ cells and $DoA = 0.0049692$.

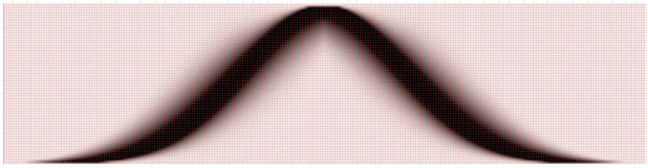

Fig. 4. An example of grid type-2 fuzzy set.

B. Operations

This section is to mention operations of grid type-2 fuzzy sets involving join, meet and negation operation. For theoretic operations, join/meet operations are described as the formulas (4), (5). Suppose more than one calculation of u and w gives the same point $u \vee w$, for example, $u_1 \vee w_1 = \theta^*$ and $u_2 \vee w_2 = \theta^*$. Then within the computation of the formulas (4), (5), we would have

$$f_x(u_1) \star g_x(w_1)/\theta^* + f_x(u_2) \star g_x(w_2)/\theta^* \qquad (19)$$

where + denotes union. Combining these two terms for the common θ^* is a type-1 computation in which t-conorm can be used, e.g. the maximum.

If $\theta \in F \sqcup G$, the possible $\{u, w\}$ pairs that can give θ as the result of the maximum operation are $\{u, \theta\}$ where $u \in (-\infty, \theta]$ and $\{\theta, w\}$ where $w \in (-\infty, \theta]$. The process of finding the membership of θ in $\tilde{A} \sqcup \tilde{B}$ can be computed as follows:

$$f_{F \sqcup G}(\theta) = \phi_1(\theta) \vee \phi_2(\theta) \qquad (20)$$

where $\phi_1(\theta) = g_x(\theta) \wedge \sup_{u \in (-\infty, \theta]} \{f_x(u)\}$ (for join operation) or $\phi_1(\theta) = g_x(\theta) \wedge \sup_{u \in [\theta, 1]} \{f_x(u)\}$ (for meet operation); $\phi_2(\theta) = f_x(\theta) \wedge \sup_{w \in (-\infty, \theta]} \{g_x(w)\}$ (for join operation) or $\phi_2(\theta) = f_x(\theta) \wedge \sup_{w \in [\theta, 1]} \{g_x(w)\}$(for join operation) [4].

Let $\tilde{A}^g$, $\tilde{B}^g$ be two grid type-2 fuzzy sets that their parameters are $x^{\tilde{A}}_{min}$, $x^{\tilde{A}}_{max}$, $u^{\tilde{A}}_{min}$, $u^{\tilde{A}}_{max}$, $x^{\tilde{B}}_{min}$, $x^{\tilde{B}}_{max}$, $u^{\tilde{B}}_{min}$, $u^{\tilde{B}}_{max}$, respectively. Note that $\tilde{A}^g$, $\tilde{B}^g$ are discretized on the same grid with dx, du parameters. Results of meet/join operations of $\tilde{A}^g$, $\tilde{B}^g$, called $\tilde{C}^g$, are grid type-2 fuzzy sets with the same partitioned grid that its domain is union of domains $D(\tilde{A}^g)$ and $D(\tilde{B}^g)$. At each vertex (i, j) of the grid of $\tilde{C}^g$, membership grades are computed as follows:

$$\mu_{\tilde{C}}(x_i, u_j) = \left[f_i^{\tilde{B}^g} \wedge \mu_{\tilde{A}^g}(x_i, u_j)\right] \vee \left[f_i^{\tilde{A}^g} \wedge \mu_{\tilde{B}^g}(x_i, u_j)\right] \quad (21)$$

in which $f_i^{\tilde{G}^g} = max_{k=0}^{i} \mu_{\tilde{G}^g}(x_i, u_k)$ for join operation and $f_i^{\tilde{G}^g} = max_{k=i}^{N} \mu_{\tilde{G}^g}(x_i, u_k)$ for meet operation. $\wedge$ is t-conorm operation that may be maximum. $\vee$ is t-norm operation that may be minimum or product.

The algorithm for computing operations of grid type-2 fuzzy sets for GPU platform that uses the capability of matrix computation, are described as follows:

Algorithm 3.1: Join Operation of G-T2FS

Input: $\tilde{A}^g$, $\tilde{B}^g$ be grid type-2 fuzzy sets.

$min_{\tilde{A}^g}$, $max_{\tilde{A}^g}$, $min_{\tilde{B}^g}$, $max_{\tilde{B}^g}$ are minimum or maximum indices of G-T2FSs.

Output: $\tilde{C}^g$ be result G-T2FS.

1) Initializing device memory m_A, m_B, m_C for matrices of grid $\tilde{A}^g$, $\tilde{B}^g$, $\tilde{C}^g$, respectively.
2) Copy data from host memory to device memory of grid $\tilde{A}^g$, $\tilde{B}^g$.
3) Initializing blocks and threads for GPU. We implement with number of threads $t_x = t_u = 32$ per block. So size of grid is computed as $n_{block} = M_{\tilde{C}^g}/(t_x * k_x)$ and $m_{block} = M_{\tilde{C}^g}/(t_u * k_u)$ where k_x, k_u are size of sub-matrix processed for each thread.
4) For each thread do
   a) Set ix = blockIdx.x*blockDim.x + threadIdx.x;
   b) Set iu = blockIdx.y*blockDim.y + threadIdx.y;
   c) for $(i = ix * k_x; i < (ix + 1) * k_x; i++)$
   d) for $(j = iu * k_u; j < (iu + 1) * k_u; j++)$
      i) if $(i < min(min_{\tilde{A}^g}, min_{\tilde{B}^g})$ or $i > max(max_{\tilde{A}^g}, max_{\tilde{B}^g})$ or $(m_A[i,j] == 0$ and $m_A[i,j] == 0))$ $m_C[i,j] = 0$
      ii) else if $((i > min(min_{\tilde{A}^g}, min_{\tilde{B}^g})$ and $i < max(min_{\tilde{A}^g}, min_{\tilde{B}^g}))$ or $(i > min(max_{\tilde{A}^g}, max_{\tilde{B}^g})$ and $i < max(max_{\tilde{A}^g}, max_{\tilde{B}^g}))$ $m_C[i,j] = m_A[i,j] > 0 ? m_A[i,j] : m_B[i,j]$
      iii) else compute $m_C[i,j]$ as the formula (21).
5) Copy data from device memory to host memory of the result grid $\tilde{C}^g$.

The next is to describe algorithm for meet operation.

Algorithm 3.2: Meet Operation of G-T2FS

Input: $\tilde{A}^g$, $\tilde{B}^g$ be grid type-2 fuzzy sets.

$min_{\tilde{A}^g}$, $max_{\tilde{A}^g}$, $min_{\tilde{B}^g}$, $max_{\tilde{B}^g}$ are minimum or maximum indices of G-T2FSs.

Output: $\tilde{C}^g$ be result G-T2FS.

1) Initializing device memory m_A, m_B, m_C for matrices of grid $\tilde{A}^g$, $\tilde{B}^g$, $\tilde{C}^g$, respectively.
2) Copy data from host memory to device memory of grid $\tilde{A}^g$, $\tilde{B}^g$.
3) Initializing blocks and threads for GPU. We implement with number of threads $t_x = t_u = 32$ per block. So size of grid is computed as $n_{block} = M_{\tilde{C}^g}/(t_x * k_x)$ and $m_{block} = M_{\tilde{C}^g}/(t_u * k_u)$ where k_x, k_u are size of sub-matrix processed for each thread.
4) For each thread do
   a) Set ix = blockIdx.x*blockDim.x + threadIdx.x;
   b) Set iu = blockIdx.y*blockDim.y + threadIdx.y;
   c) for $(i = ix * k_x; i < (ix + 1) * k_x; i++)$
   d) for $(j = iu * k_u; j < (iu + 1) * k_u; j++)$
      i) if $(i < max(min_{\tilde{A}^g}, min_{\tilde{B}^g})$ or $i > min(max_{\tilde{A}^g}, max_{\tilde{B}^g})$ or $(m_A[i,j] == 0$ and

$m_A[i,j] == 0)$) $m_C[i,j] = 0$

ii) else compute $m_C[i,j]$ as the formula (21).

5) Copy data from device memory to host memory of the result grid $\tilde{C}^g$.

C. Defuzzification

This operation is to find the centroid of uncertainty of space of grid type-2 fuzzy sets. So the section mentions an algorithm to compute the 3-dimensional centroid of the grid by combining from centroids of cells. The following is outline of defuzzification algorithm:

Algorithm 3.3: Defuzzification operation

Input: $\tilde{A}^g$ is a grid type-2 fuzzy sets.

Output: value of defuzzification.

1) Initializing s_x, s_u are sizes of sub-matrix for processing of each thread.
2) Initializing device memory m_A for matrices of grid $\tilde{A}^g$.
3) Copy data from host memory to device memory of grid $\tilde{A}^g$.
4) Initializing blocks and threads for GPU. We implement with number of threads $t_x = t_u = 32$ per block. So size of grid is computed as $n_{block} = M_{\tilde{C}^g}/(t_x * k_x)$ and $m_{block} = M_{\tilde{C}^g}/(t_u * k_u)$ where k_x, k_u are size of sub-matrix processed for each thread.
5) Initializing device memory m_d, m_u for storing data for each thread.
6) Call ds be area of a cell in the grid of T2FS.
7) $f1 = f2 = 0$
8) For each thread do
   a) Set ix = blockIdx.x*blockDim.x + threadIdx.x;
   b) Set iu = blockIdx.y*blockDim.y + threadIdx.y;
   c) for $(i = ix * k_x; i < (ix + 1) * k_x; i++)$
   d) for $(j = iu * k_u; j < (iu + 1) * k_u; j++)$
      i) $fa = (m_A[i,j] + m_A[i+1,j] + m_A[i,j+1] + m_A[i+1,j+1])/4.0$.
      ii) $fu = (j + 0.5)/N$.
      iii) $f1 += fa * ds * (i + 0.5) * fu$.
      iv) $f2 += fa * ds * fu$.
   e) $m_d[ix, iu] = f1$, $m_u[ix, iu] = f2$.
9) $f1 = f2 = 0$.
10) Copy data from device memory of m_d, m_u to host memory.
11) Compute sum of $m_d[i,j]$s, call s_1, and sum of $m_u[i,j]$s, call s_2.
12) Value of defuzzification is s_1/s_2.

Hence, the algorithm is implemented on both of GPU (at threads) and CPU, results of threads are combined. Experiments of this algorithm are shown in next section.

IV. Experiments

In this section, we mention experiments on representation and operations of grid type-2 fuzzy sets involving join , meet, negation operation and defuzzification. All experiments are implemented on ASUS Laptop with CPU CORE^{TM} i5, 6GB RAM, Windows 7 64bit and nVIDIA GEFORCE GT 520M with 48 cores using VC 2008 Release Compiler and CUDA SDK 4.0. Table I shows summary of representation of $\tilde{A}$ mentioned at Example 3.1 with various parameters.

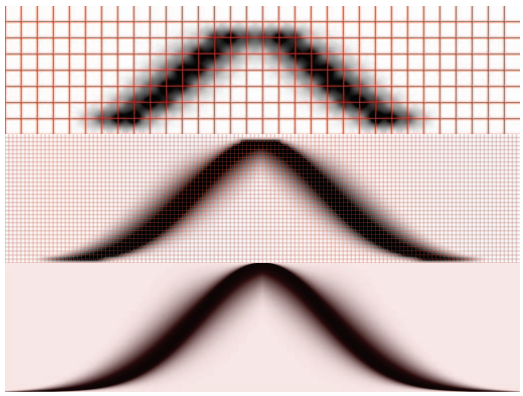

Fig. 5. Grid Type-2 Fuzzy Sets with various parameters.

TABLE I
GRID TYPE-2 FUZZY SETS WITH VARIOUS PARAMETERS

M × N	dx	du	DoA
32 × 8	0.13015658	0.14285715	0.0677241
128 × 32	0.031770505	0.032258064	0.011788
1024 × 256	3.944139e-3	3.9215689e-3	7.545143e-4
4096 × 1024	9.853124e-4	9.765625e-4	8.908835e-5

Fig. 7 depicts G-T2FS with various parameters mentioned in Table I. See that the accuracy of grid type-2 fuzzy sets depends on size of the grid.

The next is experiments of operations involving join, meet, negation. Each operation is deployed on both of CPU and GPU platform. Algorithms are implemented 10 times for taking average of run-times of each operation. Input grid type-2 fuzzy sets $\tilde{A}^g$, $\tilde{B}^g$ are the same as the example 3.1 in which parameters are set as $\sigma^{\tilde{A}^g} = 0.6, m_1^{\tilde{A}^g} = 3, m_2^{\tilde{A}^g} = 4$ and $\sigma^{\tilde{B}^g} = 0.5, m_1^{\tilde{B}^g} = 5, m_2^{\tilde{B}^g} = 6$. As summarised report in Table II, when size of grid is large, algorithms of CPU platform take huge run-times meanwhile run-times of GPU algorithms are allowable to deploy real applications with high accuracy. For low complexity operations such as negation or defuzification, difference of run-times between CPU and GPU is not large and enough fast for applications even high resolution of the grid.

Fig. 6 shows output G-T2FSs of operations with 2048×512 resolution.

Fig. 7 depicts plots of scale between CPU and GPU run-times in various resolution of the grid.

Defuzzification operation is implemented on above mentioned $\tilde{A}^g$. The truth value of defuzzification of $\tilde{A}^g$ is 3.5. Table III shows that difference between the truth value and the output is enough small in various resolution. High resolution of grid results in high accuracy of defuzzification.

TABLE II
RUNTIME OF OPERATIONS WITH VARIOUS GRIDS (IN MILLISECONDS)

Operation		$2^7 \times 2^5$	$2^9 \times 2^7$	$2^{11} \times 2^9$	$2^{13} \times 2^{10}$
Meet	CPU	0.395	24.032	1562.951	24691.961
	GPU	0.263	2.558	77.457	1088.108
	CPU/GPU	1.502	9.395	20.178	22.693
Join	CPU	0.251	10.866	665.912	10893.129
	GPU	0.824	1.689	43.128	548.333
	CPU/GPU	0.305	6.433	15.440	19.866
Negation	CPU	0.024	0.623	24.323	237.821
	GPU	0.231	0.599	7.475	47.986
	CPU/GPU	0.104	1.040	3.254	4.956

Fig. 6. Outputs of operations of G-T2FS; the top: JOIN; the left-bottom: MEET and the right-bottom:NEGATION

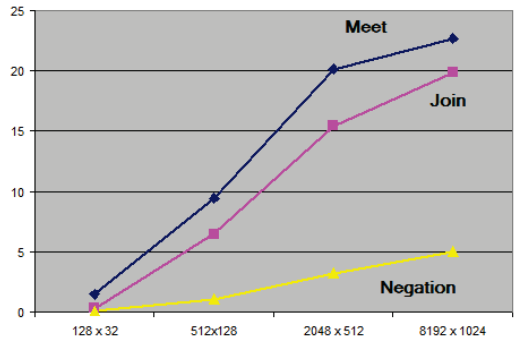

Fig. 7. Plot of scale between CPU/GPU runtime of operations.

TABLE III
RUNTIME AND DIFFERENCE VALUE OF DEFUZZIFICATION OPERATION

Difference	128×32	512×128	2048×512	8192×1024
CPU (ms)	0.067	1.075	48.161	434.821
Difference	5.5e-5	5.0e-6	4.0e-5	9.83e-4
GPU (ms)	0.75	0.755	5.202	39.272
Difference	5.81e-4	4.0e-6	7.0e-6	1.7e-5
CPU/GPU	0.088	1.433	9.259	11.072

V. CONCLUSION

The paper has presented an approach to representation of general type-2 fuzzy sets for speed-up of computation based on GPU. So concepts of grid type-2 fuzzy sets are proposed and be the basis to deploy operations of general type-2 fuzzy sets, involving join, meet, negation and defuzzification operation. These algorithms of operations are implemented on GPU platform and obtain high performance on paralleling computation in comparison with CPU platform.

These algorithms are basis to solve further issues on computation of general type-2 fuzzy sets. So the next goals is to develop G-T2FS - based approaches for inference techniques, general type-2 fuzzy logic systems, interpolative reasoning.

ACKNOWLEDGMENT

This paper is sponsored by Vietnam National Foundation for Science and Technology Development (NAFOSTED), Grant No 102.012010.12, and the Research Fund RFit@LQDTU, Faculty of Information Technology, Le Quy Don University.

REFERENCES

[1] S.Coupland, R.John, Geometric Type-1 and Type-2 Fuzzy Logic System, IEEE Trans. on Fuzzy Systems, 15(1), pp. 3-15, 2007.
[2] S.Coupland, R.John, New geometric inference techniques for type-2 fuzzy sets, Int' Journal of Approximate Reasoning, 49(1), pp. 198-211, 2008.
[3] N. Karnik, J.M. Mendel, Liang Q. Type-2 Fuzzy Logic Systems, IEEE Trans. on Fuzzy Systems, 7(6), pp. 643-658, 1999.
[4] N. Karnik, J.M. Mendel, Operations on Type-2 Fuzzy Sets, Fuzzy Sets and Systems, 122, pp.327-348, 2001.
[5] N. Karnik, J.M. Mendel, Centroid of a Type-2 Fuzzy Set, Information Sciences, 132, pp. 195-220, 2001.
[6] J.M. Mendel, R. I. John, Type-2 Fuzzy Sets Made Simple, IEEE Trans. on Fuzzy Systems, 10(2), pp. 117-127, 2002.
[7] J.M. Mendel , R.John, F. Liu, Interval Type-2 Fuzzy Logic Systems Made Simple, IEEE Trans. on Fuzzy Systems, 14(6), pp. 808-821, 2006.
[8] J. M. Mendel, F. Liu, D. Zhai, α-Plane Representation for Type-2 Fuzzy Sets: Theory and Applications, IEEE Trans. on Fuzzy Systems, 17(5), pp. 1189-1207, 2009.
[9] F. Liu, An efficient centroid type-reduction strategy for general type-2 fuzzy logic system, Infor. Sciences, vol.178(9),pp. 2224-2236, 2008.
[10] L.T.Ngo, L.T.Pham, P.H.Nguyen, K.Hirota, On approximate representation of type-2 fuzzy sets using triangulated irregular network, Foundations of Fuzzy Logic and Soft Computing,LNCS 4529, Springer:584-593,2007.
[11] L. T. Ngo, L. T. Pham, P. H. Nguyen, K.Hirota, Refinement geometric algorithms for type-2 fuzzy set operations,IEEE-FUZZ 09:866-871, 2009.
[12] L. T. Ngo, Refinement CTIN for General Type-2 Fuzzy Logic Systems, IEEE-FUZZ 2011:1225-1232, 2011.
[13] Janusz T. Starczewski, Efficient triangular type-2 fuzzy logic systems, Int' Journal of Approximate Reasoning, Vol. 12(5), pp. 799-811, 2009.
[14] H. Hagras, A Hierarchical Type-2 Fuzzy Logic Control Architecture for Autonomous Mobile Robots, IEEE Trans. on Fuzzy Systems, Vol. 12(4), pp. 524-539, 2004.
[15] H. Hagras, Type-2 FLCs: A new Generation of Fuzzy Controllers, IEEE Computational Intelligence Magazine, Vol. 2(1), pp 30-43, 2007.
[16] C.Wagner, H.Hagras, Toward General Type-2 Fuzzy Logic Systems based on zSlices, IEEE Trans.on Fuzzy Systems, vol.18(4):637-660, 2010.
[17] D.Anderson, R.Luke, J.Keller, Speed-up of Fuzzy Clustering Through Stream Processing on Graphics Processing Units, IEEE Trans. on Fuzzy Systems, Vol. 16(4), pp. 1101-1106, 2008.
[18] Anderson, Parallelisation of Fuzzy Inference on a Graphics Processor Unit Using the Compute Unified Device Architecture, The 2008 UK Workshop on Computational Intelligence, UKCI 2008, pp. 1-6, 2008.
[19] N.Harvey, R.Luke, J. Keller, D.Anderson, Speed-up of Fuzzy Logic through Stream Processing on Graphics Processing Units, Proc. IEEE Congress on Evolutionary Computation, pp.3809-3815, 2008.
[20] K.Sejun, C.Donald, A GPU based Parallel Hierarchical Fuzzy ART Clustering, Proceedings of International Joint Conference on Neural Networks, vol.1, pp.2778-2782, 2011.
[21] F.Chia, C.Teng, Y.Wei, Speed-up of Implementing Fuzzy Neural Networks With High-Dimensional Inputs Through Parallel Processing on GPU, IEEE Trans.on Fuzzy Systems, Vol.19(4):717-728, 2011.
[22] NVIDIA CUDA Programming Guide, *http : //www.nvidia.com/*