Calculation of HELAS amplitudes for QCD processes using graphics processing unit (GPU)

K. Hagiwara¹, J. Kanzaki²,a, N. Okamura²,b, D. Rainwater³, and T. Stelzer⁴,c

¹ KEK Theory Center and Sokendai, Tsukuba 305-0801, Japan
² KEK, Tsukuba 305-0801, Japan
³ Space and Geophysics Laboratory, Applied Research Laboratories, University of Texas, Austin, TX 78758, USA
⁴ Dept. of Physics, University of Illinois, Urbana, IL, USA

dated: [date] / Revised version: September 29, 2009

Abstract. We use a graphics processing unit (GPU) for fast calculations of helicity amplitudes of quark and gluon scattering processes in massless QCD. New HEGET [HELAS Evaluation with GPU Enhanced Technology] codes for gluon self-interactions are introduced, and a C++ program to convert the MadGraph generated FORTRAN codes into HEGET codes in CUDA (a C-platform for general purpose computing on GPU) is created. Because of the proliferation of the number of Feynman diagrams and the number of independent color amplitudes, the maximum number of final state jets we can evaluate on a GPU is limited to 4 for pure gluon processes ($gg \rightarrow 4g$), or 5 for processes with one or more quark lines such as $q\bar{q} \rightarrow 5g$ and $qq \rightarrow q\bar{q} + 3g$. Compared with the usual CPU-based programs, we obtain 60-100 times better performance on the GPU, except for 5-jet production processes and the $gg \rightarrow 4g$ processes for which the GPU gain over the CPU is about 20.

1 Introduction

In our previous report [1], we introduced a C-language version of the HELAS codes [3], HEGET [HELAS Evaluation with GPU Enhanced Technology], which can be used to compute helicity amplitudes on a GPU (Graphics Processing Unit). Encouraging results with 40-150 times faster computation speed over the CPU performance were obtained for pure QED processes, $q\bar{q} \rightarrow n\gamma$, for $n = 2$ to 8 in $pp$ collisions.

In this paper, we extend our study to QCD processes with massless quarks and gluons. The HEGET routines for massless quarks and gluons are identical to those of quarks and photons introduced in [1], and the $gg$ vertex function structure is also the same as the $q\gamma$ functions. The only new additional routines are those for $gg$ and $gggg$ vertices. For the QED processes studied in ref. [1], we found that the present CUDA compiler cannot process $q\bar{q} \rightarrow 6\gamma$ amplitude with $6! \approx 700$ Feynman diagrams, and we need to subdivide the HEGET codes into small pieces for $6\gamma$ and $7\gamma$ processes. In the case of 8-flavor production with $8! \approx 4 \times 10^4$ Feynman diagrams, we have not been able to compile the program even after subdivision into small pieces. We also encountered serious slow down when the program accesses global memory during the parallel processing period. Therefore, our concern for evaluating the QCD processes on a GPU is the proliferation of the number of diagrams, as well as the number of independent color amplitudes which come with different color weights.

The paper is organized as follows. In section 2, we present the cross section formula for $n$-jet production processes in $pp$ collisions in the quark-parton model, or in the leading order of perturbative QCD with scale-dependent parton distribution functions (PDF’s). In section 3, we review briefly the structure of GPU computing by using HEGET codes, and give basic parameters of the GPU and CPU machines used in this analysis. In section 4, we introduce new HEGET functions for $ggg$ and $gggg$ vertices. Section 5 gives our results and section 6 summarizes our findings. Appendix lists all the new HEGET codes introduced in section 4.

2 Physics Process

2.1 $n$-jet production in $pp$ collisions

The cross section for $n$-jet production processes can be expressed as

$$d\sigma = \sum_{\{a,b\}} \int d\alpha_a d\alpha_b D_{ab} (x_a,Q) D_{bp} (x_b,Q) d\hat{s}(\hat{s}) , \quad (1)$$

where $D_{ab}$ and $D_{bp}$ are the scale $(Q)$ dependent parton distribution functions (PDF’s), $x_a$ and $x_b$ are the momentum fractions of the partons $a$ and $b$, respectively, in the
Table 1. The number of Feynman diagrams and the color bases for QCD processes studied in this paper.

| No. of jets in the final state | $gg \rightarrow$ gluons | #diagrams | #colors | $u\bar{u} \rightarrow$ gluons | #diagrams | #colors | $uu \rightarrow uu+gluons$ | #diagrams | #colors |
|-------------------------------|-----------------------|-----------|--------|--------------------------------|-----------|--------|----------------------------|-----------|--------|
| 2                             | 6                     | 6         | 3      | 2                              | 2         | 2      | 2                          | 2         | 2      |
| 3                             | 45                    | 24        | 18     | 6                              | 10        | 8      | 4                          | 76        | 40     |
| 4                             | 510                   | 120       | 159    | 24                             | 76        | 40     | 4                          | 786       | 240    |
| 5                             | 7245                  | 720       | 1890   | 120                            | 786       | 240    |

right- and left-moving protons. For the total $pp$ collision energy of $\sqrt{s}$,

$$\hat{s} = s x_a x_b, \quad (2)$$

gives the invariant mass squared of the hard collision process

$$a + b \rightarrow 1 + 2 + \cdots + n. \quad (3)$$

The subprocess cross section is computed in the leading order as

$$d\hat{\sigma}(\hat{s}) = \frac{1}{2s} \frac{1}{2} \sum \frac{1}{n_{a,b}} \sum \left| \mathcal{M}_{\lambda_i}^{c_i} \right|^2 d\Phi_n, \quad (4)$$

where

$$d\Phi_n = (2\pi)^4 \delta^4 \left( p_a + p_b - \sum_{i=1}^n p_i \right) \prod_{i=1}^n \frac{d^3 p_i}{(2\pi)^3 2\omega_i}, \quad (5)$$

is the invariant $n$-body phase space, $\lambda_i$ are the helicities of the initial and final partons, $n_a$ and $n_b$ are the color degree of freedom of the initial partons $a$ and $b$, respectively, and $c_i$ represents the color indices of the initial and final partons. When there are more than one gluons or identical quarks in the final states, an appropriate statistical factor should be multiplied on the phase space $d\Phi_n$ in eq. (5).

The Helicity amplitudes for the process (1)

$$a(p_a, \lambda_a, c_a) + b(p_b, \lambda_b, c_b) \rightarrow 1(p_1, \lambda_1, c_1) + \cdots + n(p_n, \lambda_n, c_n) \quad (6)$$

can be expressed as

$$\mathcal{M}_{\lambda_i}^{c_i} = \sum_{l: \text{diagram}} (M_{\lambda_i})^{c_i}_l \quad (7)$$

where the summation is over all the Feynman diagrams. The subscripts $\lambda_i$ stand for a given combination of helicities ($\pm 1$ for both quarks and gluons in the HELAS convention [3]), and the subscripts $c_i$ correspond to a set of color indices (1, 2, 3 for flowing-IN quarks, $\bar{2}, 2, \bar{3}$ for flowing-OUT quarks, and 1 to 8 for gluons). In MadGraph [4] the amplitudes are expanded as

$$\mathcal{M}_{\lambda_i}^{c_i} = \sum_{\alpha} T^{c_i}_\alpha (J_{\lambda_i})_\alpha \quad (8)$$

in the color bases $T^{c_i}_\alpha$ which are made from the SU(3) generators in the fundamental representation [5] for the total $pp$ collision energy of $\sqrt{s}$.

The color factors are computed as

$$N_{\alpha\beta} = \frac{1}{N_{a,b}} \sum_{c_i} (T^{c_i}_\alpha (T^{c_i}_\beta)^*) \quad (9)$$

where $n_{a,b} = 3$ for $q$ and $\bar{q}$, $n_{a,b} = 8$ for gluons, and the summation is over all $\{c_i\} = \{c_a, c_b, c_1, \ldots, c_n\}$. The color sum-averaged square amplitudes are computed as

$$\sum_{c_i} |\mathcal{M}_{\lambda_i}^{c_i}|^2 = \sum_{a,b} (J_{\lambda_i})_a N_{a\beta} (J_{\lambda_i})_\beta^* \quad (10)$$

The cross sections are then expressed as

$$d\hat{\sigma}(\hat{s}) = \frac{1}{2s} \sum_{\lambda_i} \sum_{c_i} |\mathcal{M}_{\lambda_i}^{c_i}|^2 d\Phi_n, \quad (11)$$

where we introduce the helicity sum-average symbol as

$$\sum_{\lambda_i} = \frac{1}{22} \sum_{\lambda_i}. \quad (12)$$

In this paper the following three types of multi-jet production processes are computed:

$$gg \rightarrow 99.999.9999 \quad (13a)$$
$$q\bar{q} \rightarrow 99.999.9999.99999 \quad (13b)$$
$$uu \rightarrow uu, uug, uugg, uuggg \quad (13c)$$

The number of contributing Feynman diagrams and the number of color bases for the above processes are summarized in Table 1, which includes those for the process, $gg \rightarrow 5g$. We note here that the number of diagrams (7245) for $gg \rightarrow 5g$ exceeds that of the $u\bar{u} \rightarrow 7\gamma$ process ($7! \approx 5040$), for which we could run the converted MadGraph codes on a GPU, only after division into small pieces [1]. In fact, we have not been able to run the $gg \rightarrow 5g$ program on GPU even after dividing the program into more than 100 pieces; as explained in section 5.4.

Proliferation of the number of independent color basis vectors is also a serious concern for GPU computing, since the color matrix $N$ of eq. (9) has $m(m+1)/2$ elements when there are $m$ independent basis vectors $T^{c_i}_\alpha$. For example, the process $uu \rightarrow uugg$ has $m = 240$ color basis vectors from Table 1 and the matrix has $3 \times 10^4$ elements. The matrix exceeding 16000 elements cannot be stored in the 64kB constant memory, while storing it in the global memory will result in serious loss of efficiency in parallel computing. Therefore, the method to handle summation over color degrees of freedom is a serious concern in GPU computing.
2.2 Selection criteria for jets

Total and differential cross sections of the processes in $pp$ collisions at $\sqrt{s}=14$ TeV are computed in this paper. We introduce final state cuts for all the jets as follows:

\[
|\eta_i| < \eta_i^{\text{cut}} = 2.5, \quad (14a)
\]
\[
p_{T_{i}} > p_{T_{i}}^{\text{cut}} = 20 \text{ GeV}, \quad (14b)
\]
\[
p_{T_{ij}} > p_{T_{ij}}^{\text{cut}} = 20 \text{ GeV}, \quad (14c)
\]

where $\eta_i$ and $p_{T_i}$ are the rapidity and the transverse momentum of the $i$-th jet, respectively, in the $pp$ collisions rest frame along the right-moving ($p_z = |p|$) proton momentum direction, and $p_{T_{ij}}$ is the relative transverse momentum between the jets $i$ and $j$ defined by

\[
p_{T_{ij}} \equiv \min(p_{T_i}, p_{T_j}) \Delta R_{ij}, \quad (15a)
\]
\[
\Delta R_{ij} = \sqrt{\Delta \eta_{ij}^2 + \Delta \phi_{ij}^2}. \quad (15b)
\]

Here $\Delta R_{ij}$ measures the boost-invariant angular separation between the jets.

As for the parton distribution function (PDF), we use the set CTEQ6L1 [7] and the factorization scale is chosen to be the cut-off $p_T$ value, $Q = p_{T_i}^{\text{cut}} = 20 \text{ GeV}$. The QCD coupling constant is also fixed as

\[
\alpha_s = \alpha_s(Q = 20 \text{GeV})_{\overline{\text{MS}}} = 0.171, \quad (16)
\]

which is obtained from the $\overline{\text{MS}}$ coupling at $Q = m_Z$, $\alpha_s(m_Z)_{\overline{\text{MS}}} = 0.118$ [8] by using the NLO renormalization group equations with 5-flavors.

3 Computation on the GPU

3.1 GPU and its host PC

For the computation of the cross sections of QCD $n$-jet production processes we use the same GPU and host PC as in the previous report [1]. In particular we use a GeForce GTX280 by NVIDIA [11] with 240 processors, whose parameters are summarized in Table 2. It is controlled by a Linux PC with Fedora 8 on a CPU whose properties are summarized in Table 3.

Programs which are used for the computation of the cross sections are developed with the CUDA [2] environment introduced by NVIDIA [11] for general purpose GPU computing.

3.2 Program structure

Our program computes the total cross sections and distributions of the QCD $n$-jet production processes via the following procedure:

1. initialization of the program,
2. random number generation for multiple phase-space points \{p_a, p_b, p_1, \ldots, p_n\} and helicities \{\lambda_i\} on the CPU,

3. transfer of random numbers to the GPU,

4. generation of helicities and momenta of initial and final partons using random numbers, and compute amplitudes ($J_{\lambda_i}$) of eq. (8) for all the color bases on the GPU,

5. multiplying the amplitudes and their complex conjugate with the color matrix $N_{ab}$ of eq. (9) and summing them up as in eq. (10), and multiply the PDF’s of the incoming partons on the GPU,

6. transferring momenta and helicities for external particles, computed weights and the color summed squared amplitudes to the CPU, and

7. summing up all values to obtain the total cross section and all distributions on the CPU.

Program steps between the generation of random numbers (2) and the summation of computed cross sections (7) are repeated until we obtain sufficient statistics for the cross section and all distributions.

3.3 Color matrix calculation

In order to compute the cross sections of the QCD multi-jet production processes, multiplications of the large color matrix $N_{ab}$ of eq. (9), the vector of color-bases amplitudes ($J_{\lambda_i}$) of eq. (8) and its complex conjugate have to be performed, as in eq. (10). For large $n$-jet processes, like $gg \rightarrow 4 \text{ gluons}$, $u\bar{u} \rightarrow 5 \text{ gluons}$ and $uu \rightarrow uu + 3 \text{ gluons}$, the dimensions of color matrices exceed 100, and the number of multiplication becomes larger than $10^4$. These matrices cannot be stored in the constant memory (64kB for the GTX280; see Table 2) which is accessed in parallel, while storing them in the global memory (1GB for GTX280) results in serious slow-down of the GPU. We find that multiplications for the color-summation in eq. (10) can be reduced significantly as follows.

\begin{table}[h]
\centering
\caption{Parameters of GeForce GTX280.}
\begin{tabular}{|c|c|}
\hline
Number of multiprocessor & 30 \\
Number of core & 240 \\
Total amount of global memory [MB] & 1000 \\
Total amount of constant memory [kB] & 64 \\
Total amount of shared memory per block [kB] & 16 \\
Total number of registers available per block [kB] & 16 \\
Clock rate [GHz] & 1.30 \\
\hline
\end{tabular}
\end{table}

\begin{table}[h]
\centering
\caption{Host PC environment.}
\begin{tabular}{|c|c|}
\hline
CPU & Core2Duo 3GHz \\
L2 Cache & 6MB \\
Memory & 4GB \\
Bus Speed & 1.333GHz \\
OS & Fedora 8 (64 bit) \\
\hline
\end{tabular}
\end{table}
Table 4. Number of different non-zero elements in the color matrix of eq. (9).

| No. of jets | $gg \to$ gluons | $u\bar{u} \to$ gluons | $uu \to uu +$ gluons |
|-------------|------------------|----------------------|---------------------|
| 2           | 3                | 2                    | 2                   |
| 3           | 7                | 4                    | 7                   |
| 4           | 15               | 9                    | 19                  |
| 5           | 45               | 24                   | 60                  |

The color matrix of eq. (9) contains many elements with the same value. We count the number of different non-zero elements in the color matrix and find the results shown in Table 4. We find for instance that among the $240 \times (240+1)/2 = 28,720$ elements of the color matrix for the $uu \to uu + 3g$ process, there are only 60 unique ones.

In general, the number of different elements in the color matrix grows linearly rather than quadratically as the number of color basis vectors grows. Since the numbers in Table 4 are small enough, we can store them in the constant memory which is accessed quickly by each parallel processor.

Before we arrive at the above solution adopted in this study, we examined the possibility of summing over colors via Monte Carlo. Let us briefly report, in passing, on this exercise.

In the Monte Carlo color summation approach, we evaluate the matrix element $M_{\lambda_i}^{\lambda_f}$ for a given set of momenta $\{p_i\}$, helicities $\{\lambda_i\}$ and colors $\{c_i\}$, and sum the squared amplitudes over randomly generated sets of $\{p_i, \lambda_i, c_i\}$. This method turns out not to be efficient because in the color basis using the fundamental representation of the SU(3) generators adopted by MadGraph, most of the basis vectors $T_{\alpha_i}^a$ vanish for a given color configuration $\{c_i\}$. As an example, $gg \to 4g$ has $5! = 120$ color basis vectors (see Table 1), which take the form

$$T_{\alpha_i}^a = \text{Tr}(T^{\alpha_1}T^{\alpha_2}\cdots T^{\alpha_5})$$

for the configuration $\{c_i\} = (a_1, a_2, \ldots, a_6)$ where $a_i$ denotes the color index of the gluon $i$ taking an integer value between 1 and 8. Among the $8^6 \approx 260,000$ configurations, only 12% give non-zero values. Moreover, as many as 75% of the color configurations give vanishing results for all the 120 basis vectors. Although the efficiency can be improved by changing the color basis, we find that our solution of evaluating the exact summation over colors is superior to the Monte Carlo summation method for all the processes which we report in this paper.

4 New HEGET functions

The HEGET functions for massless quarks and gluons are the same as those introduced in the previous report [1]. The $qgg$ vertex functions are identical to the $qg\gamma$ functions of ref. [1] except for the coupling constant:

$$e Q_q \to g_s T_{ij}^a$$

for the vertex

$$\mathcal{L}_{qgg} = -g_s T_{ij}^a A_\mu^a(x) T_i(x) \gamma_\mu q_j(x)$$

(19)

where $g_s = \sqrt{4\pi\alpha_s}$ is the strong coupling constant and $T_{ij}^a$ is an SU(3) generator in the fundamental representation. For example, the $qgg$ vertex function is computed by the HEGET function $iovxx0$ as $iovxx0$(cmplx* fi, cmplx* fo, cmplx* vc, float g, cmplx vertex)

(20)

where the coupling constants are

$$g[0] = g[1] = g_s$$

(21)

following the convention of MadGraph [4] and the color amplitude is

$$- T_{ij}^a \text{(vertex)}.$$  

(22)

In the rest of this section, we introduce new HEGET functions for three-vector boson (VVV) and four-vector boson (VVVV) vertices. All the new HEGET functions are listed in Table 5 and their contents are given in Appendix. Also shown in Table 6 is the correspondence between the HEGET functions and the HELAS subroutines.

4.1 VVV: three vector boson vertex

For the $ggg$ vertex

$$\mathcal{L}_{ggg} = g_s f^{abc}(\partial^\mu A^{av}_b(x)) A_\mu^b(x) A_\nu^c(x)$$

(23)

we introduce two HEGET functions, $vvvxxx$ and $jvvox$. They correspond to HELAS subroutines, $VVVV$, and $JVVXX$, respectively, for massless particles; see Table 5.

4.1.1 $vvvxxx$

The HEGET function $vvvxxx$ (List 1 in Appendix) computes the amplitude of the VV vertex from vector boson wave functions, whether they are on-shell or off-shell. The function has the arguments:

$$vvvxxx(\text{cmplx* ga, cmplx* gb, cmplx* gc, float g, cmplx vertex})$$

(24)

where the inputs and the outputs are:

**INPUTS:**
- cmplx ga[6] wavefunction of gluon with color index, $a$
- cmplx gb[6] wavefunction of gluon with color index, $b$
- cmplx gc[6] wavefunction of gluon with color index, $c$
- float g coupling constant of VV vertex

**OUTPUTS:**
- cmplx vertex amplitude of the VV vertex

---

1 The sign of the color amplitudes (22) and (27) follows the sign of the Lagrangian terms (14) and (29), respectively. MadGraph [4] adopts the Lagrangian with the opposite sign, that is, $(g_s)$MadGraph $= -g_s$. This sign difference is absorbed by the conventions (22) and (27).
The coupling constant is
\[ g = g_s \] (26)
in the HEGET function [24], following the convention of MadGraph [4]. In order to reproduce the amplitudes associated with the $ggg$ vertex Lagrangian of eq. (23), the color factor associated with the $ggg$ vertex is $i f_{abc}$. More explicitly, the vertex amplitude for eq. (23) is
\[ i f_{abc}(\text{vertex}) \] (27)
by using the output (vertex) in eq. (24). Also note the HELAS convention [3] of using the flowing-OUT momenta and quantum numbers for all bosons.

### 4.1.2 jvvxx0

This HEGET function jvvxx0 (List 2 in Appendix) computes the off-shell vector wavefunction from the three-point gauge boson coupling in eq. (23). The vector propagator is given in the Feynman gauge for a massless vector bosons like gluons. It has the arguments:
\[ \text{jvvxx0(cmplx* ga, cmplx* gb, float g, cmplx* jvv)} \] (28)
where the inputs and the outputs are:

**Inputs:**
- cmplx ga[6]: wavefunction of gluon with color index, \( a \)
- cmplx gb[6]: wavefunction of gluon with color index, \( b \)
- float g: coupling constant of the VVV vertex

**Outputs:**
- cmplx jvv[6]: vector current \( j^\mu(gc:ga,gb) \) which has a color index, \( c \)

As in eq. (27) the color amplitude for the off-shell current is
\[ i f_{abc}(jvv). \] (30)

### 4.2 VVVV: four vector boson vertex

For the $ggggg$ vertex
\[ \mathcal{L}_{ggggg} = \frac{g_s^2}{4} f_{abc} f_{cde} A^\mu(x)A^{br}(x)A^c_{\nu}(x)A^d_{\rho}(x) \] (31)
we introduce two HEGET functions, $ggggxx$ and jgjjggx0, listed in Table 5. They correspond to HELAS subroutines, GGGX and JGGGXX, respectively, for massless particles.

#### 4.2.1 ggggxx

The HEGET function $ggggxx$ (List 3 in Appendix) computes the portion of the amplitude of the $gggg$ amplitude where the first and the third, and hence also, the second and the fourth gluon wave functions are contracted, whether the gluons are on-shell or off-shell. The function has the arguments:
\[ ggggxx(cmplx* ga, cmplx* gb, cmplx* gc, cmplx* gd, float g, cmplx vertex) \] (32)
where the inputs and the outputs are:

**Inputs:**
- cmplx ga[6]: wavefunction of gluon with color index, \( a \)
- cmplx gb[6]: wavefunction of gluon with color index, \( b \)
- cmplx gc[6]: wavefunction of gluon with color index, \( c \)
- cmplx gd[6]: wavefunction of gluon with color index, \( d \)
- float g: coupling constant of VVV vertex

**Outputs:**
- cmplx vertex amplitude of the VVV vertex

The coupling constant $gg$ for the $ggggg$ vertex is
\[ gg = g_s^2. \] (34)

In order to obtain the complete amplitude, the function must be called three times (once for each color structure) with the following permutations:
\[ ggggxx(ga,gb,gc,gd,gg,v1) \] (35a)
\[ ggggxx(ga,gc,gd,gb,gg,v2) \] (35b)
\[ ggggxx(ga,gd,gb,gc,gg,v3) \] (35c)

The color amplitudes are then expressed as
\[ f_{abc} f_{cde} (v1) + f_{ace} f_{dbe} (v2) + f_{ade} f_{bec} (v3). \] (36)

#### 4.2.2 jgjjggx0

The HEGET function jgjjggx0 (List 4 in Appendix) computes an off-shell gluon current from the four-point gluon coupling, including the gluon propagator in the Feynman gauge. It has the arguments:
\[ jgjjggx0(cmplx* ga, cmplx* gb, cmplx* gc, float gg, cmplx* jggg) \] (37)
where the inputs and the outputs are:

**Inputs:**
- `cmplx ga[6]` wavefunction of gluon with color index, `a`
- `cmplx gb[6]` wavefunction of gluon with color index, `b`
- `cmplx gc[6]` wavefunction of gluon with color index, `c`
- `float gg` coupling constants of the VVV vertex

**Outputs:**
- `cmplx jggg[6]` vector current \( j^\mu(gd:ga,gb,gc) \) which has a color index, `d`

The function \( jggx0(ga,gb,gc,gg,j1) \) computes off-shell gluon wave function with three specific color index `d` which comes along with a specific color factor. As in eq. (35) it should be called three times

\[
jggx0(ga,gb,gc,gg,j1) \quad (39a) \\
jggx0(gc,ga,gb,gg,j2) \quad (39b) \\
jggx0(gb,gc,gb,gg,j3) \quad (39c)
\]

to give the off-shell gluon with the color factor

\[
f^{abc}f^{def}(j1) + f^{ace}f^{dbe}(j2) + f^{ade}f^{bec}(j3). \quad (40)
\]

5 Results

5.1 Comparison of total cross sections

In order to validate the new HEGET functions which are introduced in this report, we compare the total cross sections of n-jet production processes computed on the GPU with those calculated by other programs which are based on the FORTRAN version of the HELAS library. We use MadGraph/MadEvent [4] and another independent FORTRAN program which uses the Monte Carlo integration program, BASES [12], as references. Due to the limited support for the double precision computation capabilities on the GPU, the whole computations with HEGET on a GTX280 are done with single precision, while the other programs with HELAS in FORTRAN compute cross sections with double precision.

For the calculation of the n-jet production cross sections we use the same physics parameters as the MadGraph/MadEvent for all programs, and the same final state cuts of eq. (14) for all processes and all programs. The parton distribution functions of CTEQ6L1 [7] and the same factorization and renormalization scales, \( Q = p_T^{\text{cut}} = 20 \text{GeV} \), are also used.

Results for the computation of the total cross sections are summarized in Tables 6 and 8 for \( gg \to 4g \) gluons, \( u\bar{u} \to 2g \) gluons and \( uu \to uu + n-jets \) as references. We find the results obtained by the HEGET functions agree with those from the other programs within the statistics of generated number of events.

We note that multi-jet events that satisfy the final state cuts of eq. (14), where all jets are in the central region in \( |\eta| < 2.5 \), and their transverse momentum about the beam direction \( \left(\eta,\phi\right) \) and among each other \( \left(\eta,\phi\right) \) greater than 20 GeV, are dominated by pure gluonic processes in Table 6. The cross sections for \( u\bar{u} \to ng \) process in Table 7 are small because of \( u\bar{u} \) annihilation. We note that the crossing-related non-annihilation processes, \( u\bar{g} \to u + (n-1)g \), have exactly the same number of diagrams and color bases, hence can be evaluated with essentially the same amount of computation time.

5.2 Comparison of the processing time

As already described in our previous report [1], we prepare two versions of the programs in the same structure for the computation of the total cross sections. One is written in CUDA, a C-based language, and can be executed on the GPU. The other is written in C and can be executed on the CPU. Using a standard C library function we measure the time between the start of the transfer of random numbers to the GPU and the end of the transfer of computed results back to the CPU.

In Fig. 1 the measured process time in \( \mu \text{sec} \) for one event of n-jet production processes is shown for the GPU (GTX280) and the CPU (Linux PC with Fedora 8). They are plotted against the number of jets in the final state. Because the process time per event on the GPU depends strongly on the number of allocated registers at the compilation by the CUDA and the size of thread blocks at the execution time, we can combine these parameters for the fastest event process time on the GPU.

The upper three lines in Fig. 1 show the event process times on the CPU. They correspond to \( gg \to n\)-jets denoted as \( gg \), \( u\bar{u} \to n\)-jets as \( u\bar{u} \) and \( uu \to uu + n\)-jets as \( uu \), respectively. For processes with small numbers of jets, e.g. \( n_{\text{jet}} = 2 \), the event process times for different processes are all around 4.5 \( \mu \text{sec} \). This is probably because they are dominated by computation steps other than the amplitude calculations, such as computations of the PDF factors and the data transfer between GPU and CPU, which are common to all physics processes. When the number of jets becomes larger, the event process time for the same number of jets in the final states is roughly proportional to the number of diagrams of each process listed in Table 1.

The lower three lines in Fig. 1 show the event process times on a GTX280. They also correspond to \( gg \to n\)-jets denoted as \( gg \), \( u\bar{u} \to n\)-jets as \( u\bar{u} \) and \( uu \to uu + n\)-jets as \( uu \), respectively. As the number of jets becomes larger, the process time on the GPU grows more rapidly than that on the CPU. For the \( n_{\text{jet}} = 4 \) case, the event process time of \( gg \to 4g \) gluons is larger than the expected time from the proportionality to the number of diagrams of the other processes, \( u\bar{u} \to 4g \) gluons and \( uu \to uu + 2g \) gluons. In other words, the event process time on GPU grows faster than what we expect from the growth of the number of Feynman diagrams.

For instance, the event process times ratio for \( gg \to 4g \) and \( gg \to 3g \) on the CPU are roughly 120 \( \mu \text{sec} / 14 \mu \text{sec} \).
The ratio of event process times between CPU and GPU for \( gg \rightarrow 4g \) and \( uu \rightarrow 4g \) are about 120 \( \mu \)sec/29 \( \mu \)sec \( \sim 4.1 \) on GPU, as compared to the ratio of the number of Feynman diagrams in Table 4, 510/159 \( \sim 3.2 \). The same applies to \( n_{\text{jet}} = 5 \) between \( uu \rightarrow 5g \) and \( uu \rightarrow uu+gg \), where Feynman diagrams have the ratio 1890/786 \( \sim 2.4 \) from Table 4 and the event process time on the CPU gives 300 \( \mu \)sec/180 \( \mu \)sec \( \sim 1.7 \), also in rough agreement.

Table 6. Total cross sections for \( gg \rightarrow \text{gluons} \) [fb].

| No. of jets | HEGET     | Bases     | MadGraph/MadEvent |
|------------|-----------|-----------|-------------------|
| 2          | 3.1929 ± 0.0010 | 3.1928 ± 0.0010 | 3.1902 ± 0.0076 \( \times 10^{11} \) |
| 3          | 2.6201 ± 0.0023 | 2.6136 ± 0.0036 | 2.6221 ± 0.0061 \( \times 10^{10} \) |
| 4          | 5.813 ± 0.020 | 5.8140 ± 0.0095 | 5.776 ± 0.034 \( \times 10^{6} \) |

Table 7. Total cross sections for \( uu \rightarrow \text{gluons} \) [fb].

| No. of jets | HEGET     | Bases     | MadGraph/MadEvent |
|------------|-----------|-----------|-------------------|
| 2          | 2.8981 ± 0.0007 | 2.8969 ± 0.0006 | 2.8991 ± 0.0073 \( \times 10^{7} \) |
| 3          | 1.8420 ± 0.0012 | 1.8388 ± 0.0018 | 1.8421 ± 0.0077 \( \times 10^{6} \) |
| 4          | 4.465 ± 0.022 | 4.496 ± 0.017 | 4.399 ± 0.038 \( \times 10^{5} \) |
| 5          | 1.566 ± 0.057 | 1.589 ± 0.018 | 1.542 ± 0.039 \( \times 10^{5} \) |

Table 8. Total cross sections for \( uu \rightarrow uu+\text{gluons} \) [fb].

| No. of jets | HEGET     | Bases     | MadGraph/MadEvent |
|------------|-----------|-----------|-------------------|
| 2          | 2.6715 ± 0.0014 | 2.6743 ± 0.0011 | 2.6689 ± 0.0047 \( \times 10^{6} \) |
| 3          | 5.897 ± 0.004 | 5.889 ± 0.010 | 5.871 ± 0.015 \( \times 10^{7} \) |
| 4          | 2.7754 ± 0.0130 | 2.7500 ± 0.0083 | 2.748 ± 0.042 \( \times 10^{7} \) |
| 5          | 1.513 ± 0.024 | 1.560 ± 0.013 | 1.513 ± 0.024 \( \times 10^{6} \) |

The ratios of event process times between CPU and GPU for \( gg \rightarrow n_{\text{jet}} \text{-jets} \) are shown in Fig. 2. Three lines correspond to \( gg \rightarrow n_{\text{jet}} \text{-jets} \) as \( uu \rightarrow n_{\text{jet}} \text{-jets} \) as \( uu \rightarrow uu+\text{gluons}+3 \text{-gluons} \), which has 240 color bases, the ratio becomes about 20. For processes with large numbers of color bases, the ratios are smaller. For \( gg \rightarrow 4 \text{gluons} \), which has 120 color bases, the ratio is about 30, and for \( uu \rightarrow uu+\text{gluons}+3 \text{-gluons} \), which has 240 color bases, the ratio becomes about 20.

5.4 Note on \( gg \rightarrow 5g \) study

Among five-jet production processes we have not been able to run the program for \( gg \rightarrow 5g \). This process has 7245 di-
follows.

...results obtained for QED multi-photon production processes at hadron colliders on a GPU [11], Graphic Processing Unit, following the encouraging results obtained for QED multi-jet production processes at hadron colliders. We have shown the results of our attempt to evaluate QCD amplitudes on GPU.

Compilation took about 90 min. on a Linux PC. The total size of the compiled program exceeds 200 MB, and we were not able to execute this compiled program on a GTX280.

9 Summary

We have shown the results of our attempt to evaluate QCD multi-jet production processes at hadron colliders on a GPU [11], Graphic Processing Unit, following the encouraging results obtained for QED multi-photon production processes in ref. [1].

Our achievements and findings may be summarized as follows.

- A new set of HEGET functions written in CUDA [2], a C-language platform developed by NVIDIA for general purpose GPU computing, are introduced to compute triple and quartic gluon vertices. The HEGET routines for massless quarks were introduced in ref. [1], and the routine for photons [1] can be used for gluons. In addition, the HEGET functions for the qqg vertex are the same as those for the qγγ vertex introduced in ref. [1].

- The HELAS amplitude code generated by MadGraph [3] is converted to a CUDA program which calls HEGET functions for the following three type of subprocesses: gg → ng (n ≤ 5), uū → ng (n ≤ 5), and uu → uu + ng (n ≤ 3).

- Summation over color degrees of freedom was performed on a GPU by identifying the same valued elements of the color matrix of eq. (9), in order to reduce the memory size.

- All the HEGET programs for up to 5 jets passed the CUDA compiler after division into small pieces. However, we could not execute the program for the process gg → 5g. Accordingly, comparisons of performance between GPU and CPU are done for the multi-jet production processes up to 5 jets, excluding the purely gluonic subprocess.

- Event process times of the GPU program on GTX280 are more than 100 times faster than the CPU program for all the processes up to 3-jets, while the gain is reduced to 60 for 4-jets with one or two quark lines, and to 30 for the purely gluonic process. It further goes down to 30 and 20 for 5-jet production processes with one and two quark lines, respectively.

- We find that one cause of the rapid loss of GPU gain over CPU as the number of jets increases is the growth in the number of color bases. GPU programs slow down for processes with larger numbers of color basis vectors, while the performance of the CPU programs is not affected much.

- All computations on the GPU were performed with single precision accuracy. A factor of 2.5 to 4 slower performance is found for double precision computation on the GPU.

Acknowledgement. We thank Johan Alwall, Qiang Li and Fabio Maltoni for stimulating discussions. This work is supported by the Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science (No. 20340064) and the National Science Foundation (No. 0757889).

Appendix A Additional HEGET functions

In the appendix, we list the HEGET functions introduced in this report. They are for the ggγ and gggg vertices which do not have counterparts in QED. Together with the HEGET functions listed in ref. [1], the quark and gluon (photon) wave functions and the qγγ(qγγ) vertices, all the QCD amplitudes can be computed on GPU.

Appendix A.1 Functions for the VVV vertex

In the appendix, we list the HEGET functions introduced in this report. They are for the ggγ and gggg vertices which do not have counterparts in QED. Together with the HEGET functions listed in ref. [1], the quark and gluon (photon) wave functions and the qγγ(qγγ) vertices, all the QCD amplitudes can be computed on GPU.

Appendix A.1 Functions for the VVV vertex

In the appendix, we list the HEGET functions introduced in this report. They are for the ggγ and gggg vertices which do not have counterparts in QED. Together with the HEGET functions listed in ref. [1], the quark and gluon (photon) wave functions and the qγγ(qγγ) vertices, all the QCD amplitudes can be computed on GPU.

Appendix A.1 Functions for the VVV vertex

In the appendix, we list the HEGET functions introduced in this report. They are for the ggγ and gggg vertices which do not have counterparts in QED. Together with the HEGET functions listed in ref. [1], the quark and gluon (photon) wave functions and the qγγ(qγγ) vertices, all the QCD amplitudes can be computed on GPU.

Appendix A.1 Functions for the VVV vertex

In the appendix, we list the HEGET functions introduced in this report. They are for the ggγ and gggg vertices which do not have counterparts in QED. Together with the HEGET functions listed in ref. [1], the quark and gluon (photon) wave functions and the qγγ(qγγ) vertices, all the QCD amplitudes can be computed on GPU.
# Appendix A.2 Functions for the VVVV vertex

## List 3. ggggxx.cu

```c
#include "cmplx.h"

__device__
void ggggxx(cmplx* ga, cmplx* gb, cmplx* gc, cmplx* gd, float gg, cmplx* jggg)
{
    jggg[0] = ga[0]*gb[0] - ga[1]*gb[1] - ga[2]*gb[2] - ga[3]*gb[3];
    jggg[1] = ga[0]*gb[1] + ga[1]*gb[0] - gb[1]*ga[2] + gb[2]*ga[1];
    jggg[2] = ga[0]*gb[2] + ga[1]*gb[3] - gb[2]*ga[3] + gb[3]*ga[2];
    jggg[3] = ga[0]*gb[3] - ga[1]*gb[2] - gb[3]*ga[2] + gb[2]*ga[3];
    float fact = gg*(1.0/q[0]*q[1]-q[2]-q[3]*q[3]);
    cmplx jggg = fact*(ga[0]*gb[0] - ga[1]*gb[1] - ga[2]*gb[2] - ga[3]*gb[3]);
    cmplx gc = fact*(ga[0]*gb[1] + ga[1]*gb[0] - gb[1]*ga[2] + gb[2]*ga[1]);
    cmplx gb = fact*(ga[0]*gb[2] - ga[1]*gb[3] - gb[2]*ga[3] + gb[3]*ga[2]);
    cmplx ga = fact*(ga[0]*gb[3] + ga[1]*gb[2] - gb[3]*ga[2] - gb[2]*ga[3]);
    return;
}
```

## List 4. jgggxx0.cu

```c
#include "cmplx.h"

__device__
void jgggxx0(cmplx* ga, cmplx* gb, cmplx* gc, float gg, cmplx* jggg)
{
    jggg[0] = ga[0]*gb[0] - ga[1]*gb[1] - ga[2]*gb[2] - ga[3]*gb[3];
    jggg[1] = ga[0]*gb[1] + ga[1]*gb[0] - gb[1]*ga[2] + gb[2]*ga[1];
    jggg[2] = ga[0]*gb[2] + ga[1]*gb[3] - gb[2]*ga[3] + gb[3]*ga[2];
    jggg[3] = ga[0]*gb[3] - ga[1]*gb[2] - gb[3]*ga[2] + gb[2]*ga[3];
    float fact = gg*(1.0/q[0]*q[1]-q[2]-q[3]*q[3]);
    cmplx jggg = fact*(ga[0]*gb[0] - ga[1]*gb[1] - ga[2]*gb[2] - ga[3]*gb[3]);
    cmplx gc = fact*(ga[0]*gb[1] + ga[1]*gb[0] - gb[1]*ga[2] + gb[2]*ga[1]);
    cmplx gb = fact*(ga[0]*gb[2] - ga[1]*gb[3] - gb[2]*ga[3] + gb[3]*ga[2]);
    cmplx ga = fact*(ga[0]*gb[3] + ga[1]*gb[2] - gb[3]*ga[2] - gb[2]*ga[3]);
    return;
}
```

## References

1. K. Hagiwara, J. Kanzaki, N. Okamura, D. Rainwater and T. Stelzer, arXiv:0908.4403
2. http://www.nvidia.com/object/cuda_home.html
3. K. Hagiwara, H. Murayama and I. Watanabe, Nucl. Phys. B367 (1991) 257; H. Murayama, I. Watanabe and K. Hagiwara, KEK-Report 91-11, 1992.
4. T. Stelzer and W. F. Long, Comput. Phys. Commun. 81 (1994) 357.
5. See e.g. M.L. Mangano and S.J. Parke, Phys. Rept. 200, 301 (1991).
6. S. Catani, Y.L. Dokshitzer, M.H. Seymour, B.R. Webber, Nucl. Phys. B406, 187 (1993)
7. CTEQ Collaboration, H.L. Lai et al., Eur. Phys. J. C12 (2000) 375.
8. C. Amsler et al. (Particle Data Group), Phys. Lett. B667, 1 (2008) and 2009 partial update for the 2010 edition.
9. F. Maltoni and T. Stelzer, JHEP 0302 (2003) 027.
10. J. Alwall, P. Demin, S. de Visser, R. Frederix, M. Herquet, F. Maltoni, T. Plehn, D. Rainwater, T. Stelzer, JHEP 0709 (2007) 028.
11. http://www.nvidia.com/page/home.html
12. S. Kawabata, Comput. Phys. Commun. 41 (1986) 127.
13. H. Plothow-Besch, Comput. Phys. Commun. 75 (1993) 396, Int. J. Mod. Phys. A10 (1995) 2901.