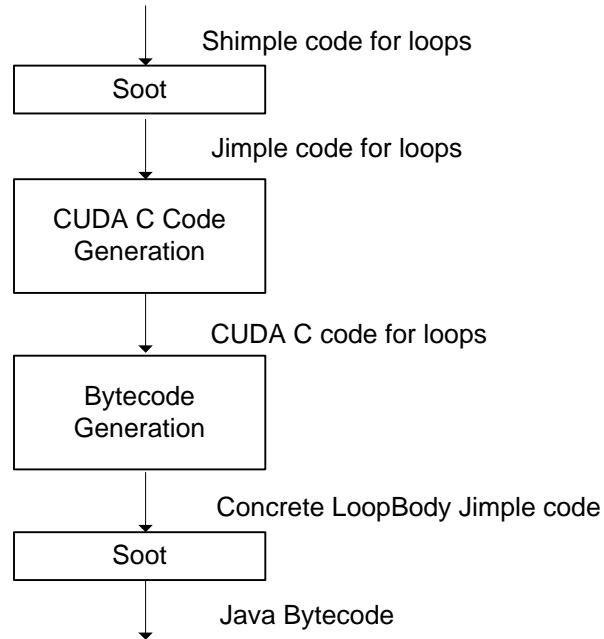


CUDA C CODE GENERATION TRANSFORMATION FLOW



APPENDIX D. COMPLETE JAVA+CUDA C VECTOR ADDITION CODE

This appendix lists all the generated and transformed code for a vector addition example. The Shimple code is listed in Java for simplicity. Figure 47 is the vector addition Java Source. Figure 48 lists the transformed vector addition Java Source. Figure 49 shows a Concrete LoopBody. Figure 50 shows a Concrete GcObjectVisitor. Finally, figure 51 displays generated CUDA C code.

```

1 package edu.syr.pcpratts.javaautogpu.runtime.test;
2
3 public class VectorAddExample {
4
5     int[] x;
6     int[] y;
7     int[] ret;
8
9     public void add(){
10        for(int i = 0; i < x.length; ++i) {
11            ret[i] = x[i]+y[i];
12        }
13    }
14 }

```

Figure 1 - Vector Addition Java Source

```

1 package edu.syr.pcpratts.javaautogpu.runtime.test;
2
3 public class VectorAddExample {
4
5     int[] x;
6     int[] y;
7     int[] ret;
8
9     public void add(){
10        QueueManager manager = QueueManager.v("0");
11        for(int i = 0; i < x.length; ++i){
12            LoopBody0 body = new LoopBody0(this, i);
13            manager.enqueue(body);
14        }
15        manager.run();
16        Iterator<LoopBody> iter = manager.iterator();
17        while(iter.hasNext()){
18            LoopBody0 curr_body = (LoopBody0) iter.next();
19            //any inter-loop dependent code below the parallel portion
20            //of the loop would be executed here. in this example there
21            //is none.
22        }
23    }
24 }

```

Figure 2- Transformed Vector Addition Java Source

```

1 package edu.syr.pcpratts.javaautogpu.generated;
2
3 import edu.syr.pcpratts.javaautogpu.runtime.LoopBody;
4 import edu.syr.pcpratts.javaautogpu.runtime.test;
5 import edu.syr.pcpratts.javaautogpu.GcObjectVisitor;
6 import edu.syr.pcpratts.javaautogpu.runtime.Memory;
7 import edu.syr.pcpratts.javaautogpu.runtime.gpu.GcHeap;

```

```

8
9 public class LoopBody0 extends LoopBody {
10
11     public VectorAddExample r0;
12     public int i0_1;
13
14     public LoopBody0(VectorAddExample r0, int i0_1){
15         this.r0 = r0;
16         this.i0_1 = i0_1;
17     }
18
19     public void run(){
20         r0.ret = r0.x[i0_1] + r0.y[i0_1];
21     }
22
23     public String getCode(){
24         StringBuilder ret = new StringBuilder();
25         //note there is a maximum size of a String in a java class file
26         //so the string is split up
27         ret.append("<compiled CUDA C code part 1>\n");
28         //...
29         ret.append("<compiled CUDA C code part N>\n");
30         return ret.toString();
31     }
32
33     public GcObjectVisitor getVisitor(Memory mem, GcHeap heap){
34         return new LoopBody0GcObjectVisitor(mem, heap);
35     }
36 }

```

Figure 3- Generated LoopBody0 Java Source

```

1 package edu.syr.pcpratts.javaautogpu.generated;
2
3 import edu.syr.pcpratts.javaautogpu.GcObjectVisitor;
4 import edu.syr.pcpratts.javaautogpu.runtime.Memory;
5 import edu.syr.pcpratts.javaautogpu.runtime.gpu.GcHeap;
6
7 public class LoopBody0GcObjectVisitor extends GcObjectVisitor {
8
9     public LoopBody0GcObjectVisitor(Memory mem, GcHeap heap){
10         super(mem, heap);
11     }
12
13     public int doWriteToHeap(Object o, boolean write_data){
14         if(o instanceof int[]){
15             return GcArrayMover.write((int[]) o, write_data);
16         }
17         if(o instanceof VectorAddExample){
18             VectorAddExample vec = (VectorAddExample) o;
19             int heap_end_ptr = mHeap.getHeapEndPtr();
20             mMem.writeByte(3);
21             mMem.writeByte(0);
22             mMem.writeByte(11);
23             mMem.writeByte(0);

```

```

24     mMem.writeInt(20);
25     mHeap.incrementHeapEndPtr(20);
26     mMem.pushAddress();
27     mMem.incrementAddress(12);
28     int ref_addr1 = writeToHeap(vec.ret, false);
29     int ref_addr2 = writeToHeap(vec.x, true);
30     int ref_addr3 = writeToHeap(vec.y, true);
31     int top_address = mMem.topAddress();
32     mMem.popAddress();
33     mMem.writeInt(ref_addr1);
34     mMem.writeInt(ref_addr2);
35     mMem.writeInt(ref_addr3);
36     mMem.setAddress(top_address);
37     return heap_end_ptr;
38 }
39 if(o instanceof LoopBody0){
40     LoopBody0 body = (LoopBody0) o;
41     int heap_end_ptr = mHeap.getHeapEndPtr();
42     mMem.writeByte(1);
43     mMem.writeByte(0);
44     mMem.writeByte(7);
45     mMem.writeByte(0);
46     mMem.writeInt(16);
47     mHeap.incrementHeapEndPtr(16);
48     mMem.pushAddress();
49     mMem.incrementAddress(8);
50     int ref_addr1 = writeToHeap(body.r0, true);
51     int top_address = mMem.topAddress();
52     mMem.popAddress();
53     mMem.writeInt(ref_addr1);
54     mMem.writeInt(body.i0_1);
55     mMem.setAddress(top_address);
56     return heap_end_ptr;
57 }
58 throw new RuntimeException("unknown type");
59 }
60
61 public Object doReadFromHeap(Object o, boolean read_data){
62     int type = readType();
63     if(type == 3){
64         return GcArrayMover.read((int[]) o);
65     }
66     if(type == 11){ //VectorAddExample
67         mMem.incrementAddress(3);
68         byte ctor_used_on_gpu = mMem.readByte();
69         if(ctor_used_on_gpu == 1){
70             o = new VectorAddExample(Sentinal.instance());
71         }
72         VectorAddExample vec = (VectorAddExample) o;
73         mMem.incrementAddress(4);
74         int address_of_ret = mMem.readInt();
75         mMem.pushAddress();
76         mMem.setAddress(address_of_ret);
77         vec.ret = readFromHeap(vec.ret);
78         mMem.popAddress();
79         mMem.incrementAddress(8);
80     return vec;

```

```

81     }
82     if(type == 7){ //LoopBody0
83         mMem.incrementAddress(3);
84         byte ctor_used_on_gpu = mMem.readByte();
85         if(ctor_used_on_gpu == 1){
86             o = new LoopBody0(Sentinal.instance());
87         }
88         LoopBody0 body = (LoopBody0) o;
89         mMem.incrementAddress(4);
90         int address_of_vec = mMem.readInt();
91         mMem.pushAddress();
92         mMem.setAddress(address_of_vec);
93         body.r0 = readFromHeap(body.r0);
94         mMem.popAddress();
95         mMem.incrementAddress(4);
96         return body;
97     }
98     throw new RuntimeException("unknown type");
99 }
100 }

```

Figure 4 - Generated LoopBody0GcObjectVisitor Java Source

```

1  __device__ int
2  edu_syr_pcpratts_gc_get_id(edu_pcpratts_gc_info gc_info){
3      return blockIdx.x * blockDim.x + threadIdx.x;
4  }
5
6  __device__ edu_pcpratts_gc_info
7  edu_syr_pcpratts_gc_init(char * gc_info_space, char * to_space,
8      char * from_space, char * to_handle_map, char * from_handle_map,
9      int to_space_free_ptr, int space_size, int cache_assoc){
10
11      mCacheAssoc = cache_assoc;
12      *((int *) (&gc_info_space[0])) = (int) to_space;
13      *((int *) (&gc_info_space[4])) = (int) from_space;
14      *((int *) (&gc_info_space[8])) = (int) to_handle_map;
15      *((int *) (&gc_info_space[12])) = (int) from_handle_map;
16      *((int *) (&gc_info_space[16])) = to_space_free_ptr;
17      *((int *) (&gc_info_space[20])) = space_size;
18
19      __syncthreads();
20
21      return gc_info_space;
22 }
23
24 __device__ char *
25 edu_syr_pcpratts_gc_deref(edu_pcpratts_gc_info gc_info, int handle){
26     char * to_space = edu_syr_pcpratts_gc_get_to_space_address(gc_info);
27     return &to_space[handle];
28 }
29
30 __device__ void
31 edu_syr_pcpratts_gc_assign(edu_pcpratts_gc_info gc_info,
32     int * lhs_ptr, int rhs){

```

```

30  *lhs_ptr = rhs;
31  }
32
33  //no cache
34  __device__ int
35  edu_syr_pcpratts_cache_get_int(edu_pcpratts_gc_info gc_info,
    int address){
36  char * to_space = (char *)
37  edu_syr_pcpratts_gc_get_to_space_address(gc_info);
38  return *((int *) &to_space[address]);
39  }
40
41  __device__ void edu_syr_pcpratts_javaautogpu_generated_LoopBody0_run(
    edu_pcpratts_gc_info gc_info, int thisref){
42  int this0 = -1;
43  int r0 = -1;
44  int i0_1;
45  int $r2 = -1;
46  int $r3 = -1;
47  int $i2;
48  int $r4 = -1;
49  int $i3;
50  int $i4;
51  edu_syr_pcpratts_gc_assign(gc_info, & this0 ,  thisref );
52
53  r0 = instance_getter_edu_syr_pcpratts_javaautogpu_generated_
    LoopBody0_r0( gc_info, this0);
54
55  i0_1 = instance_getter_edu_syr_pcpratts_javaautogpu_generated_
    LoopBody_i0_1(gc_info, this0);
56
57  $r2 = instance_getter_edu_syr_pcpratts_javaautogpu_runtime
    _test_VectorAddExample_ret(gc_info, r0);
58
59  $r3 = instance_getter_edu_syr_pcpratts_javaautogpu_runtime_test
    _VectorAddExample_x(gc_info, r0);
60  $i2 = int__array_get_cached(gc_info, $r3, i0_1);;
61
62  $r4 = instance_getter_edu_syr_pcpratts_javaautogpu_runtime_test
    _VectorAddExample_y(gc_info, r0);
63
64  $i3 = int__array_get_cached(gc_info, $r4, i0_1);;
65
66  $i4 = $i2 + $i3 ;
67  int__array_set(gc_info, $r2, i0_1, $i4 );
68  return;
69  }
70
71  __device__ void
72  edu_syr_pcpratts_javaautogpu_runtime_RuntimeBasicBlock_run(
    edu_pcpratts_gc_info gc_info, int thisref){
73
74  }
75
76  __device__ void
77  invoke_edu_syr_pcpratts_javaautogpu_generated_LoopBody0_run(

```

```

78     edu_pcpratts_gc_info gc_info, int thisref){
79     char * thisref_deref = edu_syr_pcpratts_gc_deref(gc_info, thisref);
80     GC_OBJ_TYPE_TYPE derived_type =
81     edu_syr_pcpratts_gc_get_type(thisref_deref);
82     if(0){}
83     else if(derived_type == 7){
84         edu_syr_pcpratts_javaautogpu_generated_LoopBody0_run(gc_info,
85         thisref);
86     }
87     else if(derived_type == 8){
88         edu_syr_pcpratts_javaautogpu_runtime_LoopBody0_run(gc_info,
89         thisref);
90     }
91 }
92
93 __device__ int
94 instance_getter_edu_syr_pcpratts_javaautogpu_generated_LoopBody0_r0(
95     edu_pcpratts_gc_info gc_info, int thisref){
96
97     char * thisref_deref = edu_syr_pcpratts_gc_deref(gc_info, thisref);
98     return *((int *) &thisref_deref[8]);
99 }
100
101 __device__ int
102 instance_getter_edu_syr_pcpratts_javaautogpu_generated_LoopBody0_i0_1(
103     edu_pcpratts_gc_info gc_info, int thisref){
104
105     return edu_syr_pcpratts_cache_get_int(gc_info, thisref+12);
106 }
107
108 __device__ int
109 instance_getter_edu_syr_pcpratts_javaautogpu_runtime_test_
110     VectorAddExample_ret(edu_pcpratts_gc_info gc_info, int thisref){
111
112     char * thisref_deref = edu_syr_pcpratts_gc_deref(gc_info, thisref);
113     return *((int *) &thisref_deref[8]);
114 }
115
116 __device__ int
117 instance_getter_edu_syr_pcpratts_javaautogpu_runtime_test_
118     VectorAddExample_x(edu_pcpratts_gc_info gc_info, int thisref){
119
120     return edu_syr_pcpratts_cache_get_int(gc_info, thisref+12);
121 }
122
123 __device__ int
124 instance_getter_edu_syr_pcpratts_javaautogpu_runtime_test_
125     VectorAddExample_y(edu_pcpratts_gc_info gc_info, int thisref){
126
127     return edu_syr_pcpratts_cache_get_int(gc_info, thisref+16);
128 }
129
130 __device__ void
131 instance_setter_edu_syr_pcpratts_javaautogpu_runtime_test_
132     VectorAddExample_y(edu_pcpratts_gc_info gc_info, int thisref,
133     int parameter0){
134
135     char * thisref_deref = edu_syr_pcpratts_gc_deref(gc_info, thisref);

```

```

124     edu_syr_pcpratts_gc_assign(gc_info, (int *) &thisref_deref[16],
125         parameter0);
126 }
127 __device__ int int_array_get(edu_pcpratts_gc_info gc_info,
128     int thisref, int parameter0){
129     char * thisref_deref = edu_syr_pcpratts_gc_deref(gc_info, thisref);
130     return *((int *) &thisref_deref[12+(parameter0*4)]);
131 }
132 __device__ int int_array_get_cached(edu_pcpratts_gc_info gc_info,
133     int thisref, int parameter0){
134     return edu_syr_pcpratts_cache_get_int(gc_info,
135         thisref+12+(parameter0*4));
136 }
137 __device__ void int_array_set(edu_pcpratts_gc_info gc_info,
138     int thisref, int parameter0, int parameter1){
139     char * thisref_deref = edu_syr_pcpratts_gc_deref(gc_info, thisref);
140     *((int *) &thisref_deref[12+(parameter0*4)]) = parameter1;
141 }
142 __global__ void entry(char * gc_info_space, char * to_space,
143     char * from_space, char * to_handle_map,
144     char * from_handle_map, int * handles, int * to_space_free_ptr,
145     int space_size, int cache_assoc, int iters){
146     edu_pcpratts_gc_info gc_info =
147         edu_syr_pcpratts_gc_init(gc_info_space,
148             to_space, from_space, to_handle_map, from_handle_map,
149             *to_space_free_ptr, space_size, cache_assoc);
150     int loop_control = edu_syr_pcpratts_gc_get_id(gc_info);
151     if(loop_control < iters){
152         int handle = handles[loop_control];
153         edu_syr_pcpratts_javaautogpu_generated_LoopBody0_run(gc_info,
154             handle);
155     }
156     __syncthreads();
157     *to_space_free_ptr =
158         edu_syr_pcpratts_gc_get_to_space_free_ptr(gc_info);
159     __syncthreads();
160 }

```

Figure 5 - Generated CUDA C code (abbreviated)