An Embedded C++ DSL for Object-oriented Programming with Structure-of-Arrays Layout

Matthias Springer, Hidehiko Masuhara – Tokyo Institute of Technology

SIMD (Single Instr. Mult. Data) Programming

- Structure of Arrays (SOA) Layout

```
float B_vel_x[1000], B_vel_y[1000];
float B_pos_x[1000],
      B_pos_y[1000];
float B_mass[1000];
// more fields

void B_move(int id) {
  B_pos_x[id] += t * B_vel_x[id];
  B_pos_y[id] += t * B_vel_y[id];
}

for (int i = 0; i < 1000; ++i)
    B_move(i);
```

(Automatic) Vectorization: Compiler can unroll loop and utilize SIMD instr. (e.g., load/add 4 floats)

Advantages: Good Performance

- On CPUs: *Vectorization* with SIMD Instructions
- On GPUs: *Memory Coalescing* (Combining multiple memory requests)
- On both: Good *cache utilization*
- A best practice for SIMD programming

Disadvantages: Bad Abstraction

- Code *readability* suffers
- Less *expressive* code (e.g., pointers vs. IDs)
- No C++ OOP *language features* (Constructors, methods, operators, ...)

Ikra-Cpp Programming

- AOS Style, SOA Layout

SOA Performance with OO-style Programming!


```
class Body : public SOALayout<Body, 1000> {
 public:
  IKRA_INITIALIZE_CLASS
  float_ vel_x; float_ vel_y;
  float_ pos_x; float_ pos_y;
  float_ mass;  // more fields

  void move(float t) {
    pos_x += t * vel_x;
    pos_y += t * vel_y;
  }
}; IKRA_HOST_STORAGE(Body);
```

field ID, field offset

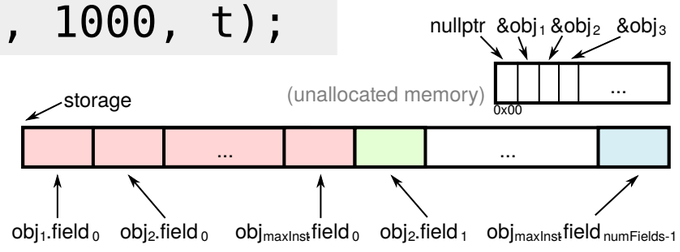
Macro/template expansion:
`Field<1, 4, float, Body> vel_y;`

Overloaded operators: Assignment, implicit conversion, member of pointer (“arrow”)

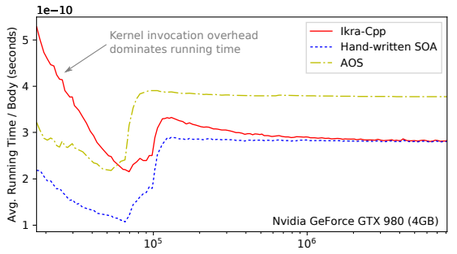
Macro/template expansion:
`char storage[Body::kSize * Body::kMaxInst]`

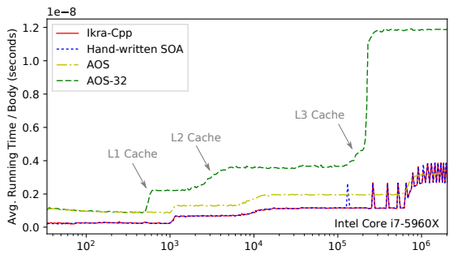
```
Body* bodies = Body::construct(1000);
execute(&Body::move, bodies, 1000, t);
```

```
Body* b = new Body();
b->vel_x = b->vel_y = 5.0;
b->move(0.1);
```



- Store all data in a statically allocated *storage buffer* in SOA layout
- *Encode object ID* in “fake pointer” (C++: `id = reinterpret_cast<uintptr_t>(ptr)`)
- Automatically decode pointers and access data with *operator overloads*
- Performance study: Same performance as hand-written SOA
- Challenge: Not breaking compiler optimizations





https://github.com/prg-titech/ikra-cpp – Google Compiler + PL Summit 2017