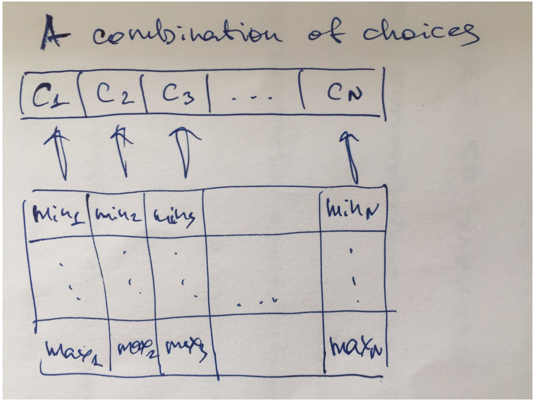
Multi-iterator Engine

Dmitry Mikushin Simon Scheidegger Philipp Eisenhauer Moritz

May 4, 2021

Multi-Iterator: the purpose


A combination of choices

Simple example

```
#include <multiit/multiit.h>

multiit::runtime::MultiIterator mi({ 2, 3, 4 });
// OR: multiit::compiletime::MultiIterator<2, 3, 4> mi;

int niters = 0;
while (1)
{
    niters++;

    bool next = mi.next();
    if (!next) break;

    const auto& current = mi.getCurrent();

    // TODO Use the current combination of choices in a target app.
}

printf(”%d iterations visited\n”, niters);
```

Supported types of multi-iterators

- `MultiIterator`: A group of indexes that iterate from 0 to the given upper value
- `LimitedMultiIterator`: A group of indexes that iterates only through indexes with total sum no greater than limit
- `GenericMultiIterator`: A group of indexes, whose indexes are themselves groups of indexes