Document Number: P1735R0
Date: 2019-06-17
Authors: Michael Wong
Project: Programming Language C++, SG19 Machine Learning
Reply to: Michael Wong <michael@codeplay.com>

# SG19: Machine Learning 2019/04/11-2019/06/13

## Contents

# Minutes for 2019/04/11 SG19 Conference Call

1.1 Roll call of participants

Michael Wong, Phil Ratzloff, David Gillies, Eugenio Bargiacchi, Frank Seide, Marco Foco, Richard Dosselmann, Steven Varga, Paul Fultv, Matthew Galati, Sebastien Messmer, Dong Ping, Rob Simpson,Gerd Heber, Mark Rankilor, Sarthak Pati

1.2 Adopt agenda

Approve

1.3 Approve minutes from previous meeting, and approve publishing previously approved minutes to ISOCPP.org

Approve

1.4 Action items from previous meetings

Michael to post Kona meeting minutes

2. Main issues (125 min)

2.1 General logistics

This is the new call slots

2 new SGs mailing lists for SG19 Machine Learning

https://groups.google.com/a/isocpp.org/forum/#!forum/sg19

and SG20 Education

https://groups.google.com/a/isocpp.org/forum/?fromgroups=#!forum/sg20

https://isocpp.org/std/forums

ML conferences
NEurIPS Dec
CVPR Long beach CA, June
ICML long beach CA, June

ICCV/ECCV

CPPCON Sept 15-20: SG19 meeting offical meeting
Meeting C++: Nov 14-16: unofficial meetup

ISO SG42 AI
https://www.iso.org/committee/6794475.html
Meeting this week in Dublin

2.2 Paper reviews

| | | | | | |
|---|---|---|---|---|---|
| P1415R0 | SG19 Machine Learning Layered List | Michael Wong, Vincent Reverdy, Ritwik Dubey, Richard Dosselmann, Eugenio Bargiacchi | 2019-01-21 | 2019-01 | SG19 |
| P1416R0 | SG19 Linear Algebra for Data Science and Machine Learning | Johann Mabille, Matthieu Brucher | 2019-01-21 | 2019-01 | SG19 |

| | | | | | |
|---|---|---|---|---|---|
| P1449R0 | Towards Tree and Graph Data Structures for C++ | Vincent Reverdy | 2019-01-21 | 2019-01 | SG19 |

Any papers proposed for review at COLOGNE?

2.2.1: ML topics
ML and scientific programming. Richard Dosselman

big data to AI
general scientific programming

http://www.gnu.org/software/gsl/
general library that encourages additional ML libraries, C focused,
experience with it in research at work
dont need everything, but may be just basic statistics, dont need fourier transform
that goes special math for ML:
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2016/p0226r0.pdf

challenge is a lot of factor, lot of properties, hard to know what to focus
BGL already have general concepts, except dynamic, but does do unidirectional,
but some concern about performance, but they have good abstractions, maybe need to modernize for C++20
compare BGL with Jgraph, also Graph500

https://graph500.org/
http://gap.cs.berkeley.edu/benchmark.html
http://dlib.net/containers.html#graph

are trees separate? maybe, but we should get graph figured out first
code examples asap
separation of data structures and algorithms

https://jgrapht.org/
both dynamic and static graph
beyond STL? have directed edges and allocations,
need to have general relation between objects
edges, need something that is a member of more then one container, bu concern about the number of allocations
STL list already has functions like splice, to move nodes between different lists, but graphs can move nodes around in general with more control

Graph Design

2.2.2 SG14 Linear Algebra progress: Bob Steagall
Different layers of proposal
https://docs.google.com/document/d/1poXfr7mUPovJC9ZQ5SDVM_1Nb6oYAXlK_d0ljdUAtSQ/edit

Apr 3 minutes? I couldn't find any.

2.2.3 any other proposal for reviews?

2.3 Other Papers and proposals

2.5 Future F2F meetings:

2.6 future C++ Standard meetings:

https://isocpp.org/std/meetings-and-participation/upcoming-meetings

- **2019-07-15 to 20: Cologne, Germany;** Nicolai Josuttis
- **2019-11-04 to 09: Belfast, Northern Ireland;** Archer Yates

- 2020-02-10 to 15: Prague, Czech Republic
- 2020-06-01 to 06: Bulgaria
- 2020-11: (New York, tentative)
- 2021-02-22 to 27: Kona, HI, USA

3. Any other business
Reflector
https://groups.google.com/a/isocpp.org/forum/#!newtopic/sg19

might move to list server

Code and proposal Staging area

4. Review

4.1 Review and approve resolutions and issues [e.g., changes to SG's working draft]

4.2 Review action items (5 min)

upload minutes from Kona
Richard to start the documents on graphs, and statistics
Marco on differentiable programming

5. Closing process

5.1 Establish next agenda

May 9

5.2 Future meeting

April 11 1-3 ET: Graph design
May 9 review Kona comments, review GG docs for graphs and Statisctics, Marco to talk differentiable programming

Jun 13: June 17 Mailing deadline
Jul 11 - cancelled? C++ Standard Meeting Cologne
Aug 8
Sep 12
Oct 10
Nov 14 - cancelled due to DST change and switching to a new cycle.

# Minutes for 2019/05/08 SG19 Conference Call

1.1 Roll call of participants
>
Frank Seide, Phil Ratzloff, David Gilles, Kirsten Lee, Marco Foco, Richard Dosselmann, Ronan Keryell, Sebastian Messmer, Michael Wong, Vincent Reverdy

>
> 1.2 Adopt agenda
>
Approve

>
> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
Approve.

> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
> All C++ reflector are now moved to listserv
>
> https://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
> 2.2 Paper reviews
>
> Any papers proposed for review at COLOGNE? Deadline June 17
>
> 2.2.1: ML topics
> Differentiable Programing by Marco Foco
>
1. wikipedia defines differentiable programming
allows automatic differentiation
used in deep learning
its reborn
2. problem is blackbox evaluation on data
feedback goes through blackbox back to generator
now optimize
synthesis and rendering needs to be done through back propagation
in general cannot back propagate through the rendering

cannot take rendering back to differentiable synthesis
3. solution 1, numeric differentiation pros and cons
exponential complexity on 2nd and 3rd order
4. solution 2 is autodiff libraries
good for 2nd order
need template black magic so heavy compile time
5.solution 3. is differentiable programming
use language primitive
more precise, use llvm-based complers doign SSA level differentiation
6. started in 1968 Maxima, graph-based
pytorch, tensorflow
VLAD
Swift, Julia, Zygote, Halide
added to Swift to say I want this function to be differentiated, will
generate optimized back prop, or can also be generated by hand
separates the kernel from the scheduling
similar in Torch, takes pyCode and converts it to SSA
Frank: this is going to make its way into all important languages soon
swift approach is with differentiable annotation and if you dont provide
the impl yourself, it can generate to see if your fn provide all the
constraints, or a custom one, Jacobian
providing your own impl only good for low level things,
power comes from building a whole NN and just ask for loss function
should be an option for when you can't provide the back prop yourself like
on GPUs, you are not likely to have the back prop
What about data sharing, with complicated fn, gradient use intermediate
results, or fn with multiple argument that you want to differentiate
Not just sharing code, but also storing intermediate results somewhere.
Agree, may be have a closure
need to use a keyword , not likely with generalized attribute
Use Herb's metacommand proposal could also do it
As a primitive (differentiate this functions) also something that modify
the AST, but we cant standardize that
Reflection is only for types, not yet for content of a function
Come from Users, only need to be light weight
Cannot be opaque
we can create a new data type with compile time templates, and can still
manage for differentiable programming, probably expression templates
Problem with Expresison template is it takes a lot of time to compile
a lot of work for the implementers, but it looks like normal code
And you still need to write generic code, and give up typing.
automatic differentiation
All feel we should move forward
and start looking at each languages solution as prior art, and then propose
our C++ solution
Frank will look at what his collegaue did with expression templates

> Richard Dosselman
> Math proposal for Machine Learning
>
>
https://docs.google.com/document/d/1VAgcyvL1riMdGz7tQIT9eTtSSfV3CoCEMWKk8GvVuFY/edit
>

Graph proposal by Richard Dosselman
address major items needed for ML
basic statistics
also good for general programming
already in Boost accumulate
python has them
use basic iterators linked list and graph data structure
when do you require from value type of iterator, any that is integrable, no strings
will this work with integers, depends on fractional, so probaby truncate and get an integer result
in matrix lib, what happens when we have eigen comes out
T would be iterator of int, but I want float
also can add default value to that template
will need to retweak it for Concepts and ranges may be in a backup session

Median can be implemented for constexpr, and string applied to a mode

allow overloads to enable to not use standard classes and predicates? yes
accumulate fn has that

get mean and standard deviation from random numbers for probability distributions

just adding a few member functions that would return those,
not hard to compute
are there distributions that dont have these values? yes like cauchy?
What is the plan for them? If we have a Distribution concept in future?
All feel this is OK to forward
do we feel we need a more generic version of that?
why not have kertosis, min, max,
if we add basic 1st 2nd moment
why not the rest, do we have justified motivation

>
>
> Graph Proposal for Machine Learning
>

>
https://docs.google.com/document/d/13rdk1Xq8ZshUiTL5QASK1N2yD5bLwK3lQjbDs5yIF6o/edit
>
> Adjacency list, property graph, forward, bidirections, and dynamic graph, multiple edges
used boost graph 7 years ago, and borrowed some of the concepts like graph traits
this breaks the compile time cyclic dependency
vertex, edge, graph type and the collection
graph_traits is the type passed to all of them

in most graph package use integer id, for the identifier for vertex, now can define a pair vertex value type as core that represnets vertex
use variety of containers to store the vertex

how flexible is this for other like adjacency arrays? all your edges are in one array,
vertexes can be stored in contguous array, but not yet a mechanism for edges
This might be worth adding
Can you build a graph over pointers, like a wrapper, instead of graph holding the edges, but just hold references
adjacency array define requirements that all edges sorted initially?Not required, just different tradeoffs
more expensive if you have to insert into middle of an array
lists and arrays have same interface but different guarantees

directed vs undirected graphs might be able to use adapters
have datalayout and code complexity goes out as you add those

vertex_set type shows performance characteristics
based on the underlying collection used
Traversals need a vertex_id on the edge to get constant time lookup

is vertex_id something the user need to know? No just an Implementation detail

static impl using array is always a question? dont want to box ourselves in so that we have to create a new data structure
this is the adjacency array or matrix (these these are 2 different things)

should we sketch a design, to see if we can see similar interface for that traversal is there an interator?
what if visitor pattern, traverse but also create new graph

Looking at Examples, Object API
can see how things are created as well as traversal
this example is just counting, but not transforming
want iterator that goes through all the edges
also want a back inserter to know what source node it came from
can you give me some examples

Algorithm Class Design
Bellman-Ford shortest path
passes weigh1
can define your own, lambda and that will retrieve a value for you

for the kind that traverse and transform, you will need a few more arguments
NO way to enforce in C++ yet , because cannot extract type of a lambda through decltype
you can do that in C++17, lambda in an unevaluated context

Algorithm class is edge_weight_fnc should be user function, else std function may have extra costs
I would use muy own type, and not a std function

Should fn parameter pass by value or ref? Good question, not sure

default argument can be std fn

continue forward, but can it make it into standard
graphs have different performance tradeoffs, am I storing vertex data next to storage array,
so its hard to generalize for

At Kona, Andrew Lumsdaine of Boost graph lib, a big problem in BGL and a functional programmign approach will help
the way different components interact, just know what not to do
as there are so many types of graphs and trees
for trees and graphs we should have independence of memory layout

defining the concepts is hard, where do you put the limit
but start with something to work with
aim for 90-95 % of problem space

please try to come up with some of those concepts
Also may be have Andrew Lumsdaine look at this
Look into adjacency array. concepts, expand edge list,

Possibly just to a tree first as graph is too complex? Surprising generic
trees are far move difficult then graphs

because trees have additional properties which needs to optimize a lot more
graphs only have 4-5 or 6 items
Going down Concept path, but not the api, trees can be used to constrain
the api, but first with graph with trees inmind.

Vincent to start a paper on trees

D1416R1: SG19 - Linear Algebra for Data Science and Machine Learning
>
> https://docs.google.com/document/d/1IKUNiUhBgRURW-UkspK7fAAyIhfXuMxjk7xKikK4Yp8/edit#heading=h.tj9hitg7dbtr
>
> P1415: Machine Learning Layered list
>
>
https://docs.google.com/document/d/1elNFdIXWoetbxjO1OKol_Wj8fyi4Z4hogfj5tLVSj64/edit#heading=h.tj9hitg7dbtr
>
> 2.2.2 SG14 Linear Algebra progress: Bob Steagall
> Different layers of proposal
>
>
https://docs.google.com/document/d/1poXfr7mUPovJC9ZQ5SDVM_1Nb6oYAXlK_d0ljdUAtSQ/edit
>
>
> 2.2.3 any other proposal for reviews?
>
>
>
>
>
> 2.3 Other Papers and proposals
>
>
> 2.5 Future F2F meetings:
>
> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
>
> - *2019-07-15 to 20: Cologne, Germany; *Nicolai Josuttis
> - *2019-11-04 to 09: Belfast, Northern Ireland;* Archer Yates
>
> - 2020-02-10 to 15: Prague, Czech Republic
- 2020-06-01 to 06: Bulgaria

- 2020-11: (New York, tentative)
- 2021-02-22 to 27: Kona, HI, USA

> 3. Any other business
>
> New reflector
>
> http://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
> Old Reflector
> https://groups.google.com/a/isocpp.org/forum/#!newtopic/sg19
> <https://groups.google.com/a/isocpp.org/forum/?fromgroups=#!forum/sg14>
>
> Code and proposal Staging area
>
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]
>
> 4.2 Review action items (5 min)
>
>
> 5. Closing process
>
>
> 5.1 Establish next agenda
>
> June 13
>
>
> 5.2 Future meeting
> April 11 1-3 ET: Graph design
> May 9
> Jun 13: June 17 Mailing deadline
> Jul 11 - cancelled? C++ Standard Meeting Cologne
> Aug 8
> Sep 12
> Oct 10
> Nov 14 - cancelled due to DST change and switching to a new cycle.

# Minutes for 2019/06/13 SG19 Conference Call

1.1 Roll call of participants
>
Michael, Matthew Galati, Jens Maurer, Mateusz Nowak, Richard Dosselmann, Kirsten Lee, David Gillies, Phil Ratzloff, Marco Foco, Vincent Reverdy

1.2 Adopt agenda
>
Approve.

> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
Approve

> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
> All C++ reflector are now moved to listserv
>
> https://lists.isocpp.org/mailman/listinfo.cgi/sg19
>

Cologne meeting F2F: propose Friday afternoon again for SG19 (Friday morning for SG14 Linear Algebra)

>
> 2.2 Paper reviews
>
> Any papers proposed for review at COLOGNE? Deadline June 17
>
DG: underlying GPU description
Affinity
https://github.com/codeplaysoftware/standards-proposals/blob/master/affinity/cpp-23/d1437r0.md
Executors
SG1

SG14: Linear Algebra

> 2.2.1: ML topics
> Differentiable Programing by Marco Foco
>
https://docs.google.com/document/d/1_5TJCBvI6fZSdyuK7_Cpo5XwdoIS24DDbEPh2JjqQsg/edit?ts=5cf823e2#heading=h.t1fxx74w88nq

Added Max
how to perform automatic differentiation: uses information from computation
to achieve same result as symbolic, but this tends to explode inside
can use tmpl expr to analyze the tree
can use intermediate computation, can use variables and does not repeat
expression, put CSE into variable and lessen explosion
Julia describes how they did it using SSA language
is this for core language? yes
follow constexpr: initally minimal expression that differentiates
expressions, say what you cant use in these expression
constexpr went with a negative list, though a positive list may be simpler
this needs a lot of narrative, motivation, to fight the perception that it
is narrowly targeted, why compiler implementer would expend effort, can
this benefit non-ML, lots of code that never heard of or use differentiation
is this a foundation building block?
can this ride on reflection? These need to analyze the AST, and not just
reflect information, we need AST manipulation, analyze the out parameters,
how do u differentiate reinterpret cast? Many things make no sense to
differentiate
Need a lot of examples of what transformation would look like f->f' what is
the partial derivative look like
for the scientific world, this is useful but we need to serve the wider
industry
need an implementer to help, like Daveed,
a reference implementation will be needed, Max might be able
add all the information in the paper, link to slide in reference
Could this be done with expr templ? first layer, then ask for core change
in later layers? Yes I explain why not Library solutions
The problem with tmpl is we loose the types, and types are needed in
creating and modifying the AST, differentiate an expr, needs a generic
lambda, so inject sampling variables, please explain this in paper to allow
other people to suggest different techniques
Phipps, Eric T <etphipp_at_[hidden]> of Kokkos interested

> Richard Dosselmann
>
>
https://docs.google.com/document/d/1VAgcyvL1riMdGz7tQIT9eTtSSfV3CoCEMWKk8GvVuFY/edit
>

added lots of eaxmples, mean, std dev
how is this not accumulate and adding after the fact? underneath is how that is done
build Tony tables, before and after
any advantages other then being common enough?
Boost accumulate has these
should we add geometric or harmonic means?
median: how to deal with non-numeric type, are they comparable
binary op why? override the operator
but you need a comparator... OK
need both examples for comparator and binary op usage
Mode: also works for nonnumerics
when mode is not unique? python throws exception
Does input has to be sorted? change example if not
make sorting same as what is in unique, same for median
can we return the range of the most used elements? Yes then you get location as well, and if there are differences, then you have access, like the address
still need equality comparator
std dev: can we add math formula to the paper
look at Random number section that shows the math
latex can gen pdf, and you can add pdf
returning std dev and not variance? any one want the variance? (just square it yourself if needed)
need to add note on precision, look at std accumulate as an example
usually dont have default arguments in algo section, just overloads so more overloads is not bad
any other basic stats? covariance for correlation, kurtosis from boost accumulate, ok to ignore
future works section
need to reorder s1 and s2
Prob distribution: P1450 from Vincent needed ... do people feel this is needed?
intended to generate random number
for normal distribution, mean and std dev are parameter
this may be a separate paper on mean and std dev, do it in a different layer

> Graph Proposal for Machine Learning
>
>
https://docs.google.com/document/d/13rdk1Xq8ZshUiTL5QASK1N2yD5bLwK3lQjbDs5yIF6o/edit
>
Phil
want adjacency list, array, matrix, bidirections and undirected graphs
adjacency list use link list for incoming and outgoing edges, so we can

iterate all of the edges all at once
add a layering plan for future proposals
functional interface based on Ranges spec
could this interface be used for previous proposal as well?
they are not integrated with ranges terminology, using a pair of iterators,
use ranges types instead
how will algo work on forward type of graph, then rewrite for the same
thing for the undirected? Like having different names?
out_edge_list argues for a functional interface, begin is not stl style
why it has no constructor parameter? due to larger memory requirement
can

conditional<graph_value_needs_wrap<EV>::value, graph_value<EV>, EV>::type;
this be conditional .. T
too many template parameters? may be not the right level of abstraction?
Should we have number of different classes?
also consider functional interface for shortest path
motivation why a class? does it store state ?
erase does not allow templ type deductions for any of the many templ
parameters
how does this relate to boost graph library; uses a concept-like mechanism
to generalize, vs very specific types in this proposal
when BGL was reviewed 7 years ago, found somethings missing,
suggest we compare with existing solution, on BGL; BGL was an interface on
top of existing impl, which may add complexity
my concern was the property lookup was column based so needed array indexing
Can we remove specific types when we plug in graphs into algo? Not sure
SImilar to STL containers and Algo using iterators
DFS and BFS match what BGL had? Why does it take the specific Graph type,
why not mine? I see, like it
But how do I transition from one object to edges? we need to find out what
mechanism, can help optimize this, like pointers?
Andrew Lumsdaine & Jens Maurer are both urging we focus on Algorithms first
eventually this should be conceptified
need concept for what is a vertex and what is an edge, and these will be
contained in data structures
how are they related, how to go from one to another, also dynamic generated
transitive closure of graph as an algorithm example
diff algo has different requirement on what it wants to do to the graph:
evaluate weighs, or insert edges,
e.g. enumerate edges from a vertex, using a range based for loop using out
and in edges

Vincent on trees

hfinkel_at_[hidden]

> D1416R1: SG19 - Linear Algebra for Data Science and Machine Learning
>
> https://docs.google.com/document/d/1IKUNiUhBgRURW-UkspK7fAAyIhfXuMxjk7xKikK4Yp8/edit#heading=h.tj9hitg7dbtr
>
> P1415: Machine Learning Layered list
>
>
https://docs.google.com/document/d/1elNFdIXWoetbxjO1OKol_Wj8fyi4Z4hogfj5tLVSj64/edit#heading=h.tj9hitg7dbtr
>
> 2.2.2 SG14 Linear Algebra progress: Bob Steagall
> Different layers of proposal
>
>
https://docs.google.com/document/d/1poXfr7mUPovJC9ZQ5SDVM_1Nb6oYAXlK_d0ljdUAtSQ/edit
>
> 2.2.3 any other proposal for reviews?
>
> 2.3 Other Papers and proposals
>
> 2.5 Future F2F meetings:
>
> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
> - *2019-07-15 to 20: Cologne, Germany; *Nicolai Josuttis
> - *2019-11-04 to 09: Belfast, Northern Ireland;* Archer Yates
>
> -2020-02-10 to 15: Prague, Czech Republic
>
> - 2020-06-01 to 06: Bulgaria
> - 2020-11: (New York, tentative)
> - 2021-02-22 to 27: Kona, HI, USA
>
> 3. Any other business
>
> New reflector
>
> http://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
> Old Reflector
> https://groups.google.com/a/isocpp.org/forum/#!newtopic/sg19
> <https://groups.google.com/a/isocpp.org/forum/?fromgroups=#!forum/sg14>
>

> Code and proposal Staging area
>
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]
>
> 4.2 Review action items (5 min)
>
> 5. Closing process
>
> 5.1 Establish next agenda
>
> TBD
>
> 5.2 Future meeting
> April 11 1-3 ET: Graph design
> May 9
> Jun 13: June 17 Mailing deadline
> Jul 11 - cancelled? C++ Standard Meeting Cologne
> Aug 8
> Sep 12
> Oct 10
> Nov 14 - cancelled due to DST change and switching to a new cycle.