ALGØRITHM SƐLƐCTIØN

Conor Hoekstra

code_report

Algorithms + Data Structures = Programs

NVIDIA RAPIDS

code_report

YouTube

https://rapids.ai

https://www.youtube.com/codereport

https://www.adspthepodcast.com

https://www.meetup.com/Programming-Languages-Toronto-Meetup/

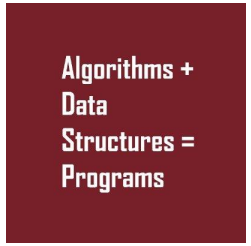
https://codereport.github.io/Algorithm-Selection/

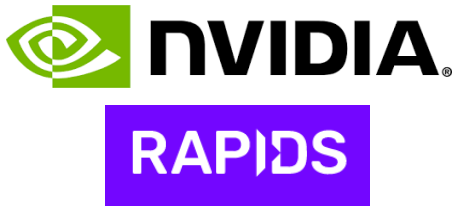
https://rapids.ai

https://www.youtube.com/codereport

https://www.adspthepodcast.com

https://www.meetup.com/Programming-Languages-Toronto-Meetup/





The code_report Blog

A blog about programming languages and more

Home Search About

Algorithm Selection

March 7, 2021

Yesterday, a contributor to one of my open source projects (Robert) pointed out to me that `std::find_if(f, l, pred) != l` is just `std::any_of(f, l, pred)`. I totally missed this while refactoring even though it is implicitly covered in my STL Algorithm Cheatsheet:

READ MORE

Meeting C++ 2020 Trip Report

November 14, 2020

This was my second time attending Meeting C++. The first time I attended was in 2019, when I gave my first Meeting C++ presentation, Better Algorithm Intuition.

READ MORE

https://www.youtube.com/watch?v=LMmFpOhcQhA

THE STL ALGORITHM CHEAT SHEET

by @code_report

ZIP ALGORITHMS

inner_product	zip_reduce
transform_reduce[17]	zip_reduce
transform	zip_with
mismatch	zip_find_not
equal	zip_reduce*

ORDER LOGN ALGORITHMS

binary_search
lower_bound
upper_bound
equal_range
partition_point

CODE REVIEW A

sort	O(nlogn)
partial_sort	O(n) – O(n^2)
nth_element	O(n)

CODE REVIEW B

find_if	O(n)
lower_bound	O(logn)

ALGORITHM RELATIONSHIPS

is_sorted -> is_sorted_until -> adjacent_find -> mismatch

THE ALGORITHM INTUITION TABLE

Algorithm	Indexes Viewed	Accumulator	Reduce / Map	Default Op
accumulate reduce[17]	1	Yes	Reduce	plus{}
	count, count_if, min_element, max_element, minmax_element			
partial_sum inclusive_scan[17]	1	Yes	Map	plus{}
find_if	1	No	Reduce	-
	find, all_of, any_of, none_of			
transform	1/2	No	Map	-
	replace[17], replace_if[17]			
adjacent_difference	2	No	Map	minus{}
inner_product transform_reduce[17]	1/2	Yes	Reduce	plus{} multiplies{}
transform_inclusive_scan[17]	1/2	Yes	Map	-
mismatch	1/2	No	Reduce	equal{}
adjacent_find	2	No	Reduce	equal{}

Note: non-accumulator reductions all short-circuit

THE TWIN ALGORITHMS

to be announced (at a future conference)

https://github.com/codereport/Algorithms

THE STL ALGORITHM CHEAT SHEET

by @code_report

ZIP ALGORITHMS

inner_product	zip_reduce
transform_reduce[17]	zip_reduce
transform	zip_with
mismatch	zip_find_not
equal	zip_reduce*

ORDER LOGN ALGORITHMS

binary_search
lower_bound
upper_bound
equal_range
partition_point

CODE REVIEW A

sort	O(nlogn)
partial_sort	O(n) – O(n^2)
nth_element	O(n)

CODE REVIEW B

find_if	O(n)
lower_bound	O(logn)

ALGORITHM RELATIONSHIPS

is_sorted -> is_sorted_until -> adjacent_find -> mismatch

THE ALGORITHM INTUITION TABLE

Algorithm	Indexes Viewed	Accumulator	Reduce / Map	Default Op
accumulate reduce[17]	1	Yes	Reduce	plus{}
	count, count_if, min_element, max_element, minmax_element			
partial_sum inclusive_scan[17]	1	Yes	Map	plus{}
find_if	1	No	Reduce	-
	find, all_of, any_of, none_of			
transform	1/2	No	Map	-
	replace[17], replace_if[17]			
adjacent_difference	2	No	Map	minus{}
inner_product transform_reduce[17]	1/2	Yes	Reduce	plus{} multiplies{}
transform_inclusive_scan[17]	1/2	Yes	Map	-
mismatch	1/2	No	Reduce	equal{}
adjacent_find	2	No	Reduce	equal{}

Note: non-accumulator reductions all short-circuit

THE TWIN ALGORITHMS

to be announced (at a future conference)

THE STL ALGORITHM CHEAT SHEET

by @code_report

ZIP ALGORITHMS

inner_product	zip_reduce
transform_reduce[17]	zip_reduce
transform	zip_with
mismatch	zip_find_not
equal	zip_reduce*

ORDER LOGN ALGORITHMS

binary_search
lower_bound
upper_bound
equal_range
partition_point

CODE REVIEW A

sort	O(nlogn)
partial_sort	O(n) - O(n^2)
nth_element	O(n)

CODE REVIEW B

find_if	O(n)
lower_bound	O(logn)

ALGORITHM RELATIONSHIPS

is_sorted -> is_sorted_until -> adjacent_find -> mismatch

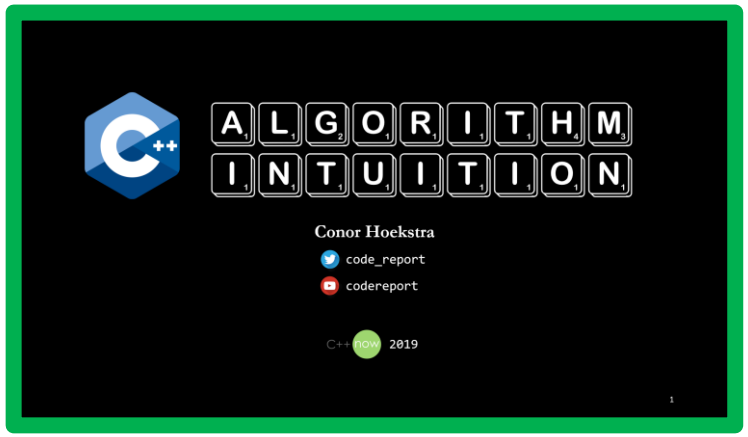
THE ALGORITHM INTUITION TABLE

Algorithm	Indexes Viewed	Accumulator	Reduce / Map	Default Op
accumulate reduce[17]	1	Yes	Reduce	plus{}
	count, count_if, min_element, max_element, minmax_element			
partial_sum inclusive_scan[17]	1	Yes	Map	plus{}
find_if	1	No	Reduce	-
	find, all_of, any_of, none_of			
transform	1/2	No	Map	-
	replace[17], replace_if[17]			
adjacent_difference	2	No	Map	minus{}
inner_product transform_reduce[17]	1/2	Yes	Reduce	plus{} multiplies{}
transform_inclusive_scan[17]	1/2	Yes	Map	-
mismatch	1/2	No	Reduce	equal{}
adjacent_find	2	No	Reduce	equal{}

Note: non-accumulator reductions all short-circuit

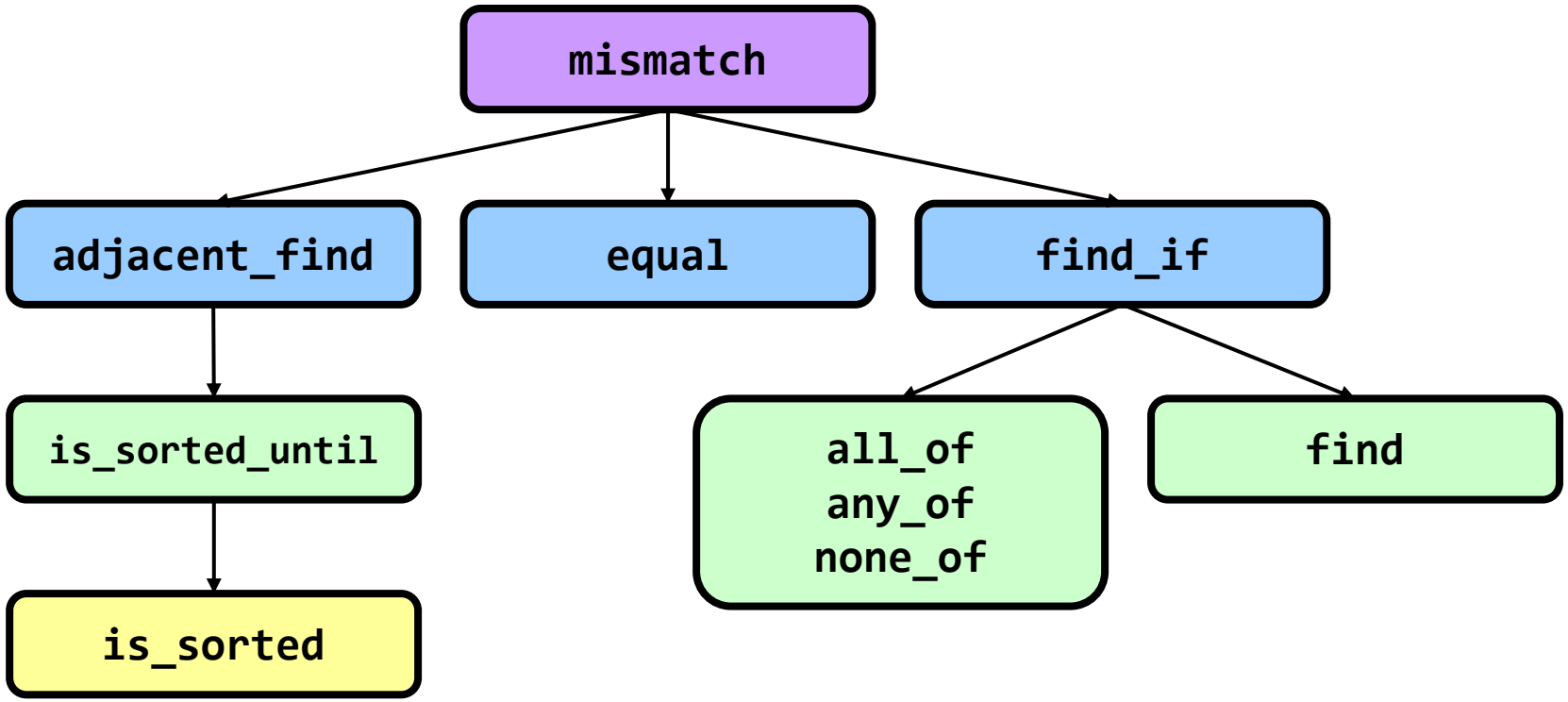
THE TWIN ALGORITHMS

to be announced (at a future conference)



Algorithm	Indexes Viewed	Accumulator	Reduce / Map	Default Op
accumulate reduce[17]	1	Yes	Reduce	plus{}
	count, count_if, min_element, max_element, minmax_element			
partial_sum inclusive_scan[17]	1	Yes	Map	plus{}
find_if	1	No	Reduce	-
	find, all_of, any_of, none_of			
transform	1/2	No	Map	-
	replace[17], replace_if[17]			
adjacent_difference	2	No	Map	minus{}
inner_product transform_reduce[17]	1/2	Yes	Reduce	plus{} multiplies{}
transform_inclusive_scan[17]	1/2	Yes	Map	-
mismatch	1/2	No	Reduce	equal{}
adjacent_find	2	No	Reduce	equal{}

Note: non-accumulator reductions all short-circuit

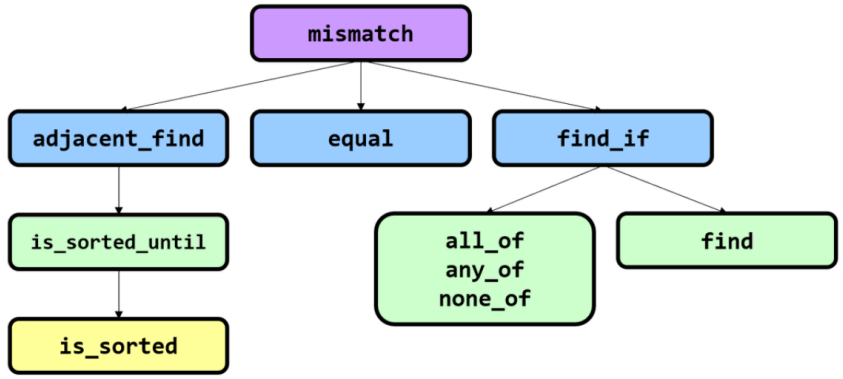

mismatch
adjacent_find
equal
find_if
is_sorted_until
all_of
any_of
none_of
find
is_sorted

```
auto equal(auto f, auto l, auto f2) {
    return std::mismatch(f, l, f2, std::not_equal_to{}).first == l;
}
```

```
auto any_of(auto f, auto l, auto pred) {
    return std::find_if(f, l, pred) != l;
}
```

Conor Hoekstra @code_report · Mar 7, 2021

Always choose the most specialized #algorithm (furthest from mismatch) @algo_love_club #cpp codereport.github.io/Algorithm-Sele...

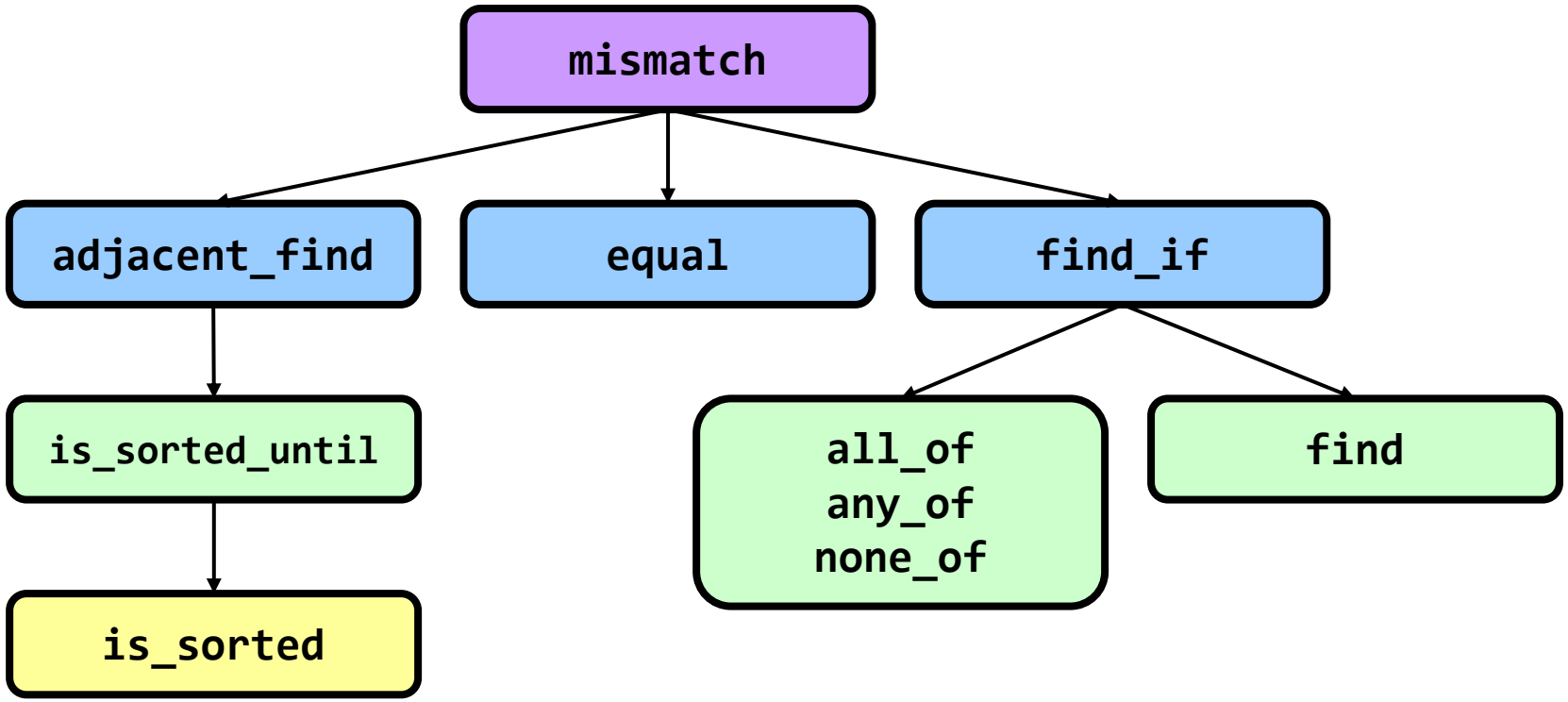


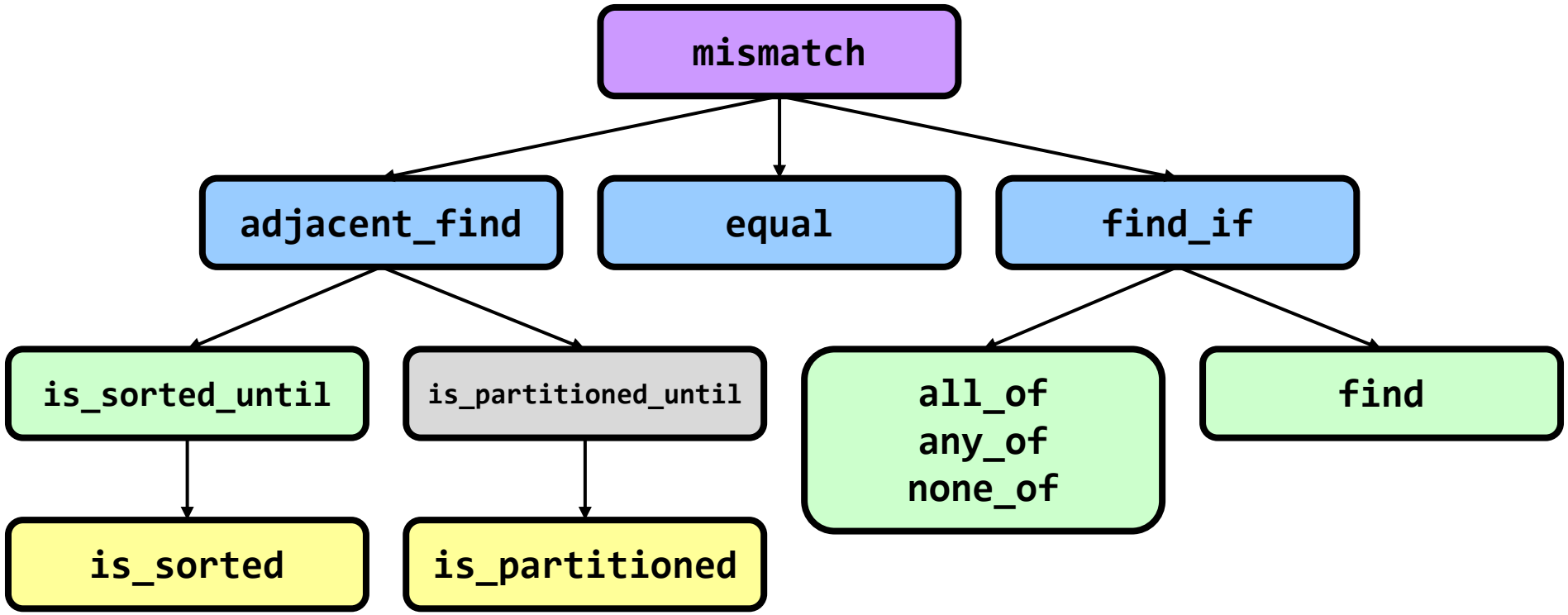
Stephan T. Lavavej
@StephanTLavavej

is_partitioned_until and is_partitioned could be added to the diagram, as they can be thought of as weaker versions of is_sorted.

6:14 PM · Mar 7, 2021

17 1 Copy link to Tweet


mismatch
adjacent_find
equal
find_if
is_sorted_until
all_of
any_of
none_of
find
is_sorted


mismatch
adjacent_find
equal
find_if
is_sorted_until
is_partitioned_until
all_of
any_of
none_of
find
is_sorted
is_partitioned

Finally, I have received a couple comments on Reddit and Twitter asking “Why?” or stating that I did not provide any reasoning. That was an oversight on my part.

> *The motivation for choosing the most specialized algorithm is that it leads to simpler and more readable code.*

```
std::find_if(f, l, pred) != l;

std::any_of(f, l, pred);
```

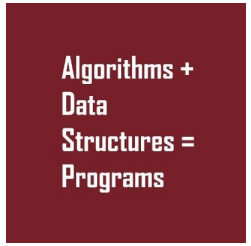
https://codereport.github.io/Algorithm-Selection/

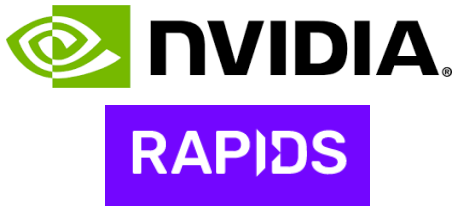
https://rapids.ai

https://www.youtube.com/codereport

https://www.adspthepodcast.com

https://www.meetup.com/Programming-Languages-Toronto-Meetup/





⌽≡⊢

https://www.dyalog.com/apl-seeds-user-meetings/aplseeds21.htm

WED, MAR 31, 8:00 AM EDT

APL Conference (for Beginners)

Online event

```
palindrome ← ⌽≡⊢
allEqual   ← ∧/(⊃=⊢)
average    ← +/÷≠
```

This is a ** FREE ** APL conference for beginners (put on by Dyalog [not me], I am just promoting it here).

...

15 attendees

Manage

Going

ALGϕRITHMS AS A TϕϕL ϕF THϕUGHT

Conor Hoekstra

code_report

NVIDIA RAPIDS #include

NVIDIA RAPIDS #include

THANK YΦU!

https://codereport.github.io/Algorithm-Selection/

https://rapids.ai

https://www.youtube.com/codereport

https://www.adspthepodcast.com

https://www.meetup.com/Programming-Languages-Toronto-Meetup/

Algorithms + Data Structures = Programs