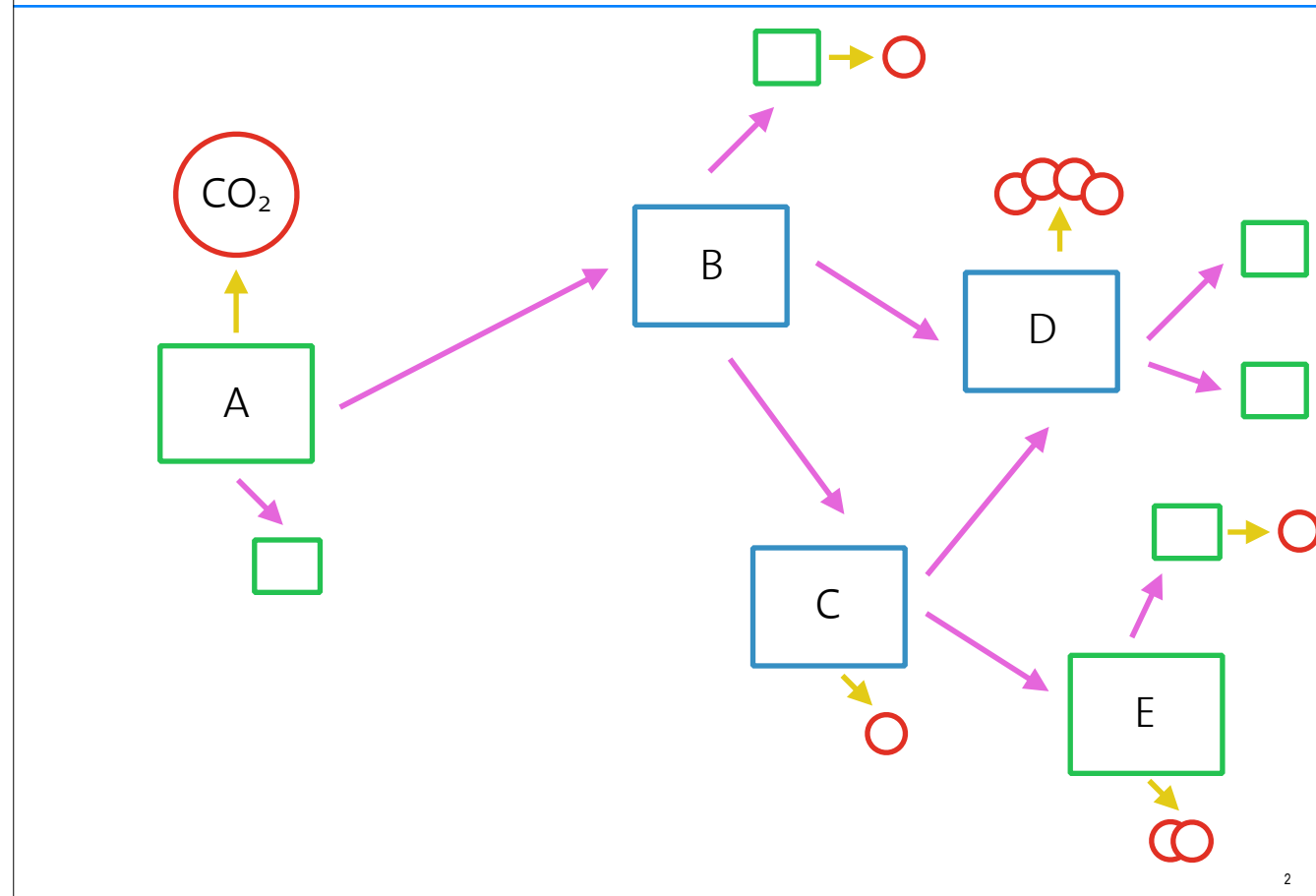




Wir schaffen Wissen – heute für morgen

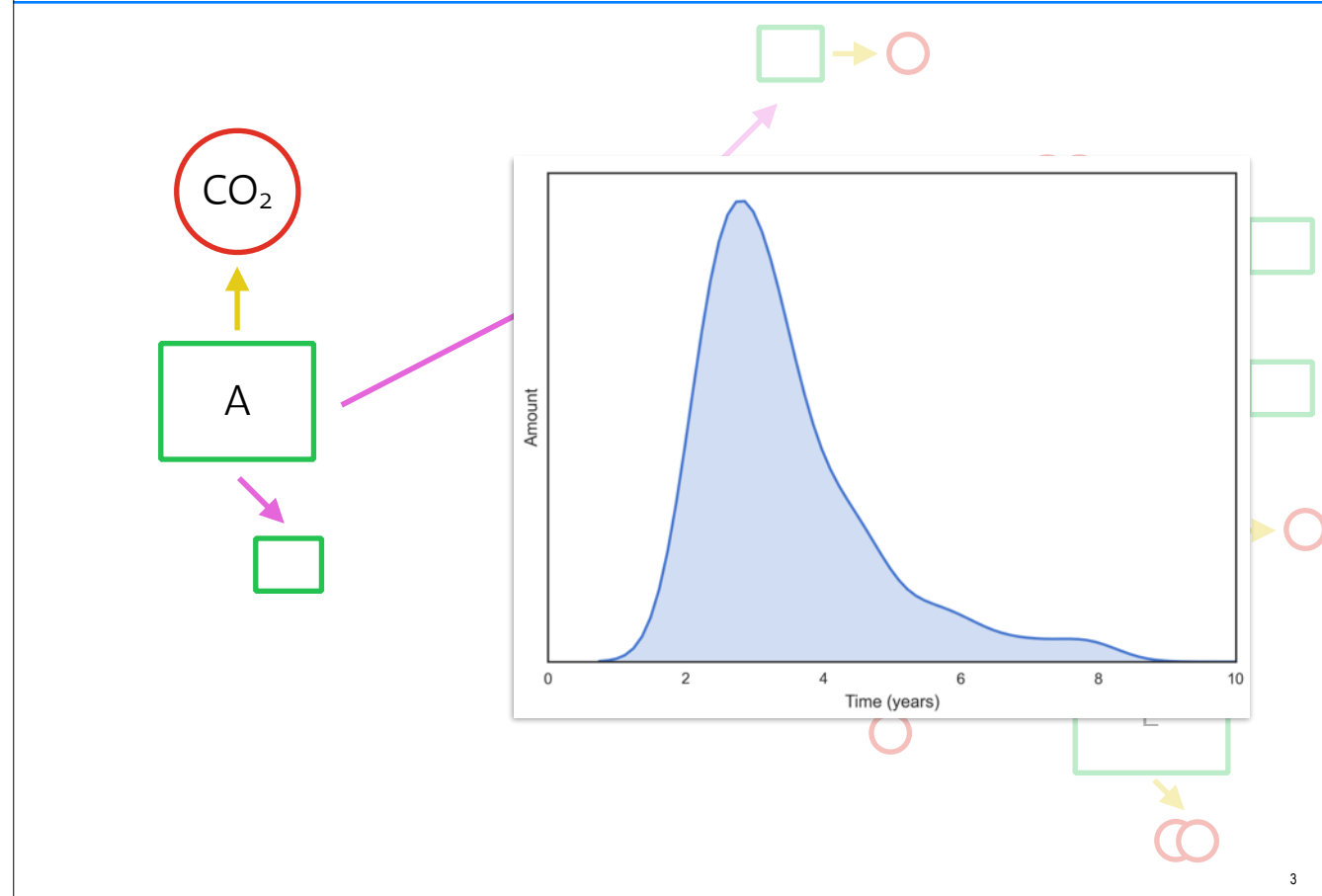
Paul Scherrer Institut
Chris Mutel
Brightway2-temporalis
A simple approach to temporal LCA calculations

Characteristics of dynamic LCA



Traditionally LCA can be conceptualised as a graph, an adjacency matrix, or a linear algebra problem - they were the same thing, as all exchanges were strictly linear.

Characteristics of dynamic LCA

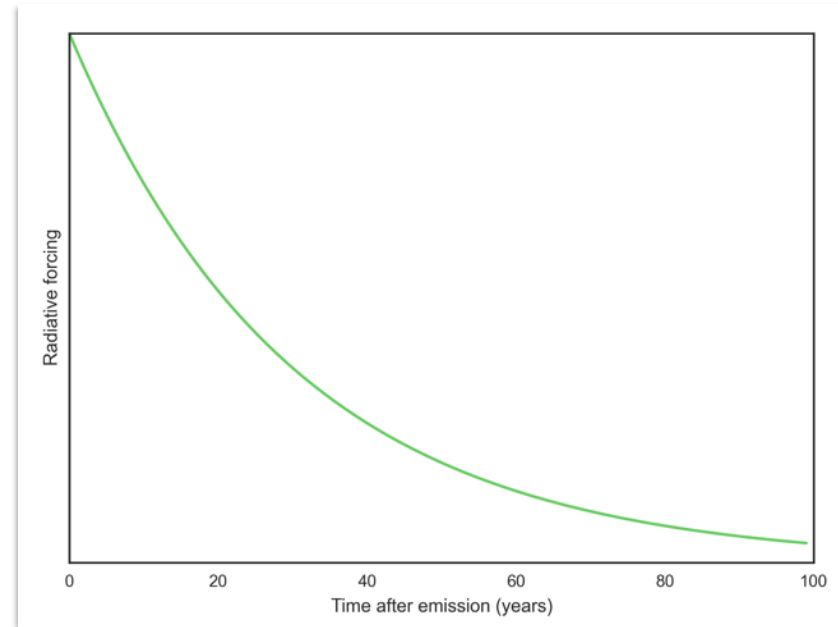
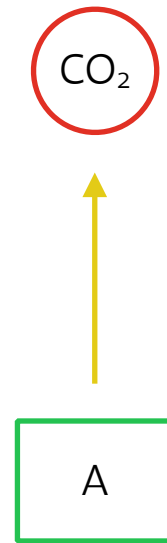


However, dynamic LCA breaks one key assumption - exchanges are still linear, but are distributed in time. For example, infrastructure construction is not instantaneous, but rather occurs for years before the unit process; similarly, emissions could continue for years after initial product manufacturing.

What do we want?	When do we want it?
Specify temporal dynamics of supply chain links	Without arbitrary temporal resolution

When we design dynamic LCA algorithms, we want to allow for flexibility, while still building a system we can understand and solve in some reasonable period of time. So, a first criteria is that a dynamic LCA algorithm should allow temporal exchanges throughout the supply chain, including biosphere exchanges. We also don't want to be forced to use any arbitrary temporal resolution, e.g. annual values; we should be able to choose the timescale that best fits each exchange.

Characteristics of dynamic LCA



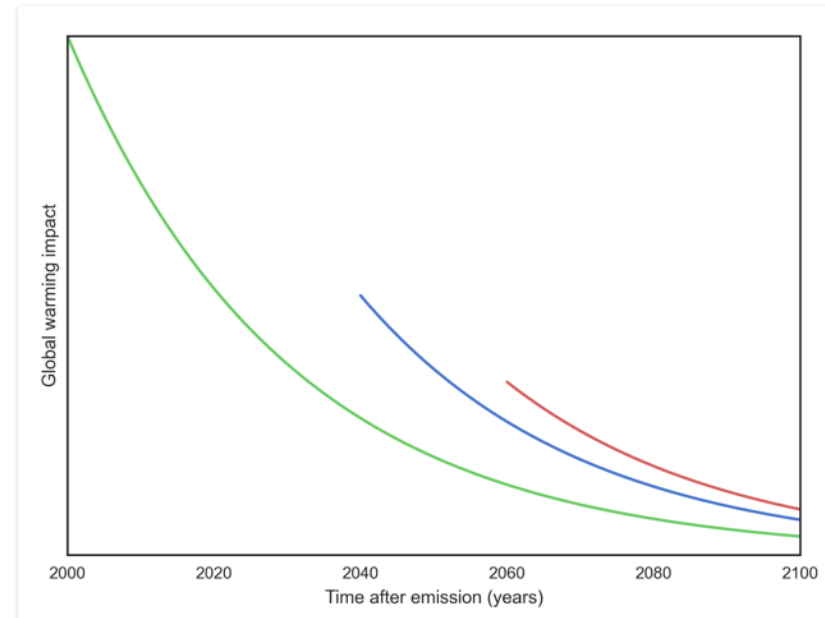
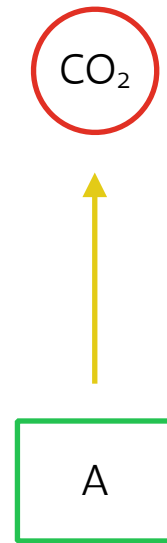
Secondly, we want to be able to spread our characterisation factor throughout time as well - global warming potential is the most well known example here, but many other impacts are spread through time as well, and indeed this information can be very helpful in providing decision support.

What do we want?	When do we want it?
Specify temporal dynamics of supply chain links	Without arbitrary temporal resolution
Distribute impacts of emissions in time	In a computationally efficient manner

6

We need to be able to combine these differing temporal resolutions (including CFs) quickly, as our supply chains are extensive; e.g. ecoinvent 3.1 is ~250,000 technosphere exchanges, ~10,000 unit processes.

Characteristics of dynamic LCA



Finally, we also want to be able to change the **total** CF depending on time of emission. Most CFs are marginal, which means that they depend on a certain background level of exposure, which can change over time. Although it adds complexity, we sometimes want to be able to include this dynamic, as things like climate urgency are real considerations for policy makers.

What do we want?	When do we want it?
Specify temporal dynamics of supply chain links	Without arbitrary temporal resolution
Distribute impacts of emissions in time	In a computationally efficient manner
Change CF magnitude based on date of emission (urgency)	Define with functions or data arrays

8

We also want to be able to specify these temporal distributions in either discretised arrays or analytical functions.

What do we want?

When do we want it?

Specify temporal dynamics of supply chain links

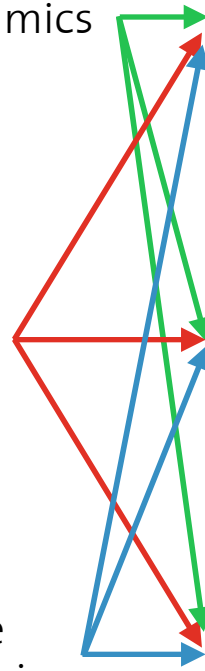
Without arbitrary temporal resolution

Distribute impacts of emissions in time

In a computationally efficient manner

Change CF magnitude based on date of emission (urgency)

Define with functions or data arrays



And of course, all our needs link to all our objectives.

Challenges & Solutions

- Data formats

```
{  
  "type": "biosphere",  
  "amount": 87,  
  "input": ("biosphere", "co2"),  
  "input": ("ecoinvent", "some process"),  
  "temporal distribution": [  
    (0, 50),  
    (1, 20),  
    (2, 10),  
    (3, 5),  
    (4, 2)  
  ]  
}
```

10

The data format used in Brightway2-temporalis: You can specify a temporal distribution in a numeric array. The first column is time, in years (but this can fractions of years as well - no restriction to integers). The second column is the amount in that time step.

Challenges & Solutions

- Data formats

```
In [1]: def make_it_funky():
        return [(x, 10 - x) for x in range(10)]

my_exchange = {
    "type": "biosphere",
    "amount": sum([x[1] for x in make_it_funky()]),
    "input": ("biosphere", "co2"),
    "input": ("ecoinvent", "some process"),
    "temporal distribution": make_it_funky()
}

my_exchange

Out[1]: {'amount': 55,
        'input': ('ecoinvent', 'some process'),
        'temporal distribution': [(0, 10),
        (1, 9),
        (2, 8),
        (3, 7),
        (4, 6),
        (5, 5),
        (6, 4),
        (7, 3),
        (8, 2),
        (9, 1)],
        'type': 'biosphere'}
```

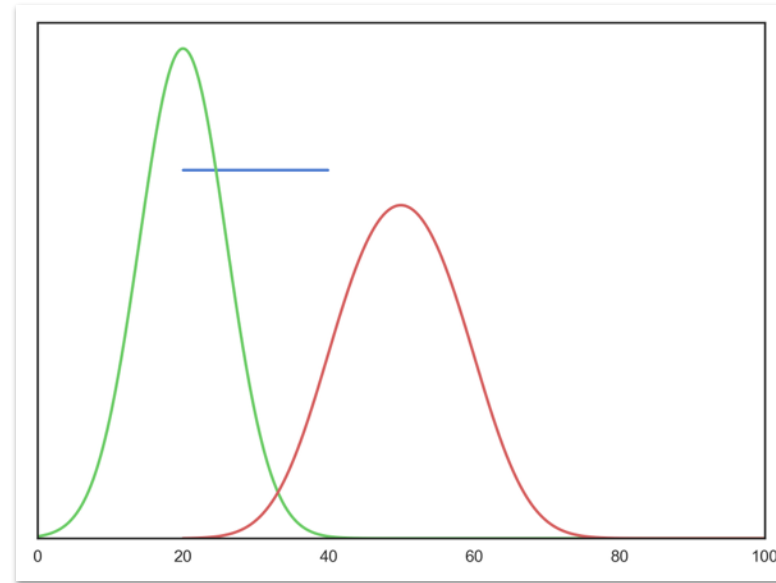
11

You can also define functions that will produce numeric arrays. This is often more elegant. Note that the serialisation format will save the function as computer code, so it can be loaded and modified later. Evaluation of the function only happens at LCA calculation.

Challenges & Solutions

- Data formats
- Computational efficiency

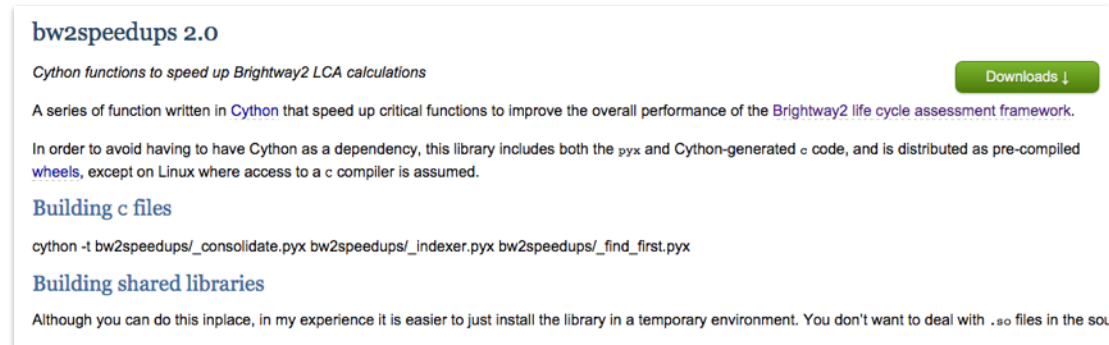
"Convolution"



Convolution is the process of combining two distributions. It sounds fancy, but isn't really - it works exactly how you think it should.

Challenges & Solutions

- Data formats
- Computational efficiency
 - Extension written in C (cython)
 - Distributions don't have to have the same temporal resolution!



bw2speedups 2.0

Cython functions to speed up Brightway2 LCA calculations

[Downloads ↓](#)

A series of function written in [Cython](#) that speed up critical functions to improve the overall performance of the [Brightway2 life cycle assessment framework](#).

In order to avoid having to have Cython as a dependency, this library includes both the `pyx` and Cython-generated `c` code, and is distributed as pre-compiled [wheels](#), except on Linux where access to a `c` compiler is assumed.

Building c files

```
cython -t bw2speedups/_consolidate.pyx bw2speedups/_indexer.pyx bw2speedups/_find_first.pyx
```

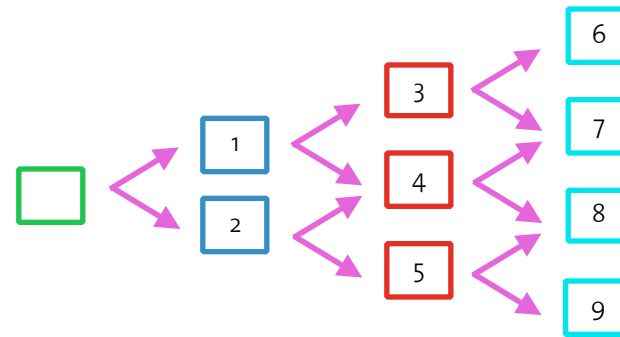
Building shared libraries

Although you can do this inplace, in my experience it is easier to just install the library in a temporary environment. You don't want to deal with `.so` files in the sou

Brightway2-speedups contains an efficient implementation of convolution that allows arbitrary timescales for both distributions.

Challenges & Solutions

- Data formats
- Computational efficiency
- Smart graph traversal



Breadth first

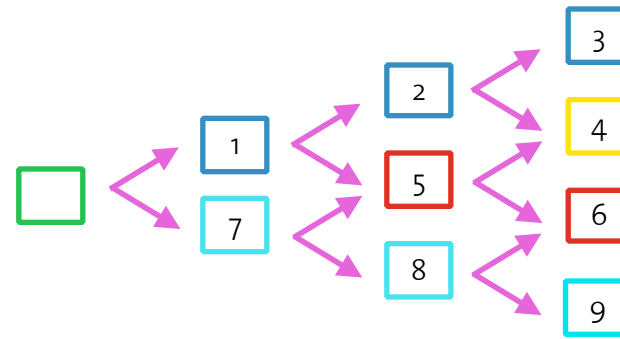
- Power series expansion
- ESPA

There are many ways to traverse the supply chain, and lots of graph theory on this subject. One main algorithm is breadth first, starting from the functional unit. The numbers show the order that each node (temporal exchange to a unit process or biosphere flow) is evaluated.

Breadth-first graph traversal is basically the same thing as power series expansion, as each level of the supply chain is evaluated sequentially.

Challenges & Solutions

- Data formats
- Computational efficiency
- Smart graph traversal

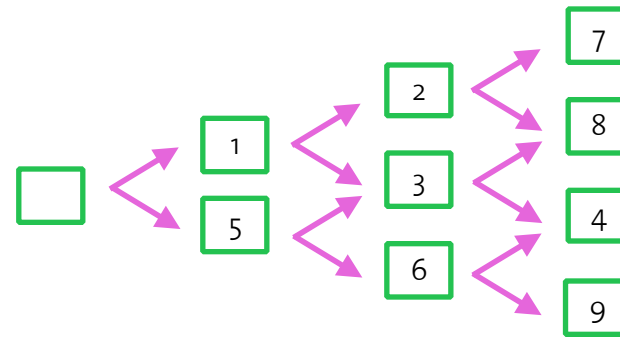


Depth first

Depth first instead goes down the supply chain instead of across. This doesn't work as well in LCA, as graphs are not directed (i.e. they have loops), and can be quite deep.

Challenges & Solutions

- Data formats
- Computational efficiency
- Smart graph traversal

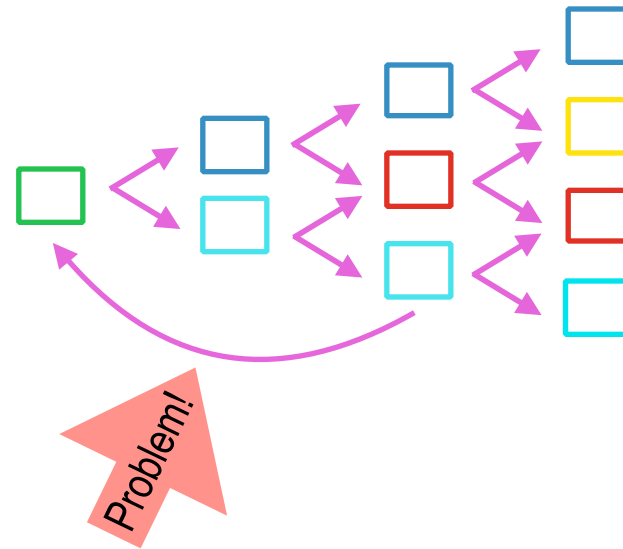


Importance first

Instead, brightway2-temporalis uses “importance-first” - each exchange is evaluated for how much of the maximum possible LCA score it could contribute, and the most potentially damaging exchanges are evaluated first. This is actually quite easy to implement, but is LCA-specific (as you need to do an LCA calculation at each node). Certainly this idea is well known among computer scientists, though I wouldn’t know where to find it or what it would be called.

Challenges & Solutions

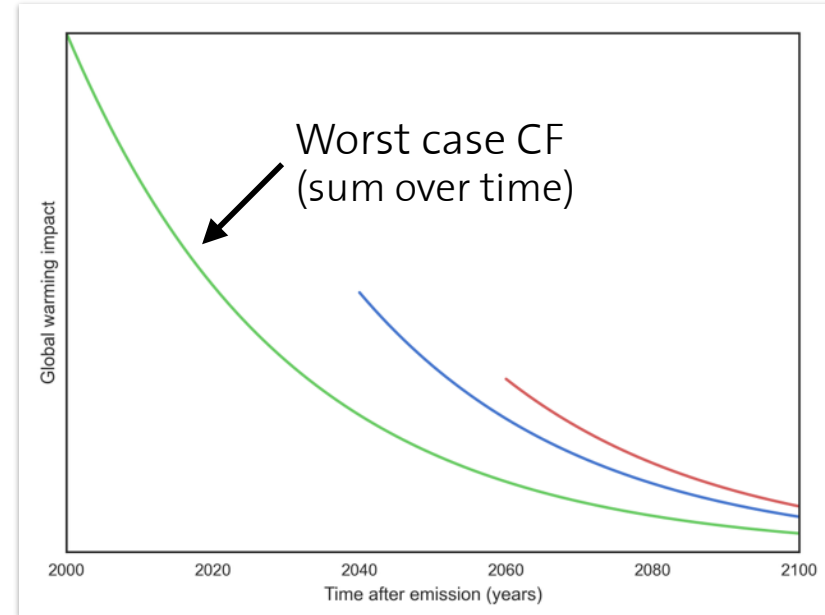
- Data formats
- Computational efficiency
- Smart graph traversal



We have to be smart when there are cycles in our graph, as naive traversal would continue ad infinitum...

Challenges & Solutions

- Data formats
- Computational efficiency
- Smart graph traversal

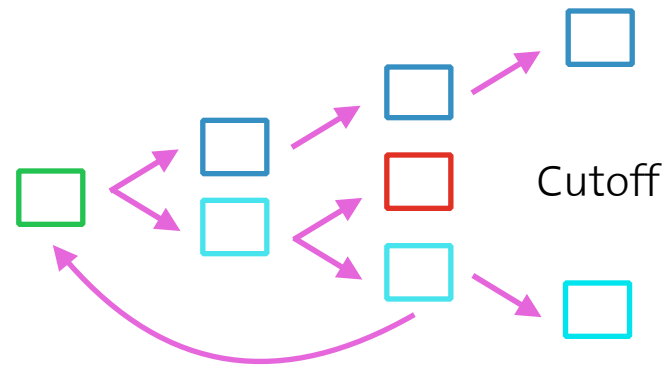


18

We can evaluate the potential contribution to the worst possible LCA score by calculating the highest CF value over time (remember, we allow our CFs total value to vary based on time of emission).

Challenges & Solutions

- Data formats
- Computational efficiency
- Smart graph traversal



We can then cutoff if the worst possible score for a certain node is much much less than the worst possible score. Of course, we need to be smart here, and default to the conservative position where we would rather do too many evaluations instead of cutting off prematurely.

Challenges & Solutions

- Data formats
- Computational efficiency
- Smart graph traversal
- Meaningful results

Timeline class

Series of points with:

Datetime

Process dataset

Biosphere flow

Amount

Easy filtering or manipulation:

```
[point for point in timeline if point.flow == ("biosphere", "co2")]
```

```
[point for point in timeline if point.dt > arrow.get("2020")]
```

Separate copy of data for characterised emissions

Finally, a little dessert: The output of the dynamic calculation is a timeline: a set of biosphere flow amount in time, each connected to a unit process. We can then apply different characterisations, and slice and dice this timeline to display useful information by flow, unit process, or time.

More information:

- Email: cmutel@gmail.com
- Website: brightwaylca.org, brightway2-temporalis.readthedocs.org



LCA people don't get to play with any of the cool toys in the foreground of this picture :(