Deleting Vertices and Relations in the `Posets` Julia Package

Deleting Vertices

Deleting vertices from a poset is somewhat different from deleting vertices in a graph. When a vertex is deleted from a graph, the vertex and edges incident with that vertex are removed. Similarly, when a vertex is removed from a poset, the relations between all the remaining vertices remain unchanged. However, this is not the same as simply deleting a vertex and its edges from the poset's Hasse diagram.

Consider the poset on the left in Figure 1. Mathematically, deleting vertex 2 from this poset results in the poset in the middle of the figure. In the Hasse diagram we have an edge from 1 upward to 3 because $1 \prec 3$ in the original poset, and so we still have $1 \prec 3$ after the deletion. Similarly, the relation $4 \prec 3$ is preserved.

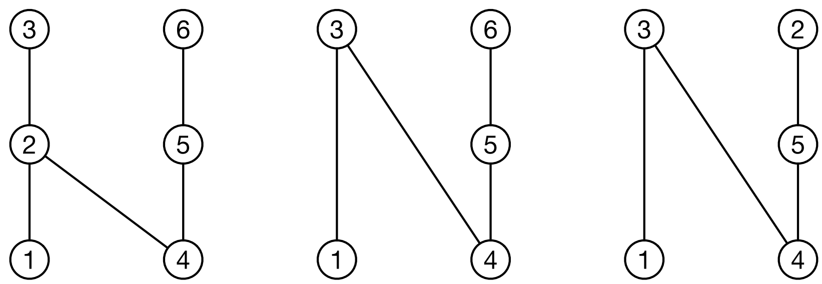

Figure 1: Vertex deletion in `Posets`. When vertex 2 is deleted from the poset on the left, mathematically we have the poset in the middle. However, following `Graphs`' convention, vertex 6 is renamed by the deleted vertex, 2, giving the result on the right.

However, the Julia `Posets` package is based on the `Graphs` package. A key convention of graphs (and hence of posets) in these packages is that the vertex set is always of the form $\{1, 2, \ldots, n\}$. If vertex n is deleted, no special action needs to be taken. But if a vertex k (with $k < n$) is removed, then the name n is no longer valid by the naming convention. In this case, the vertex formally named n is renamed k (the label of the deleted vertex). As a result, the result in `Posets` of deleting vertex 2 in Figure 1 is the poset on the right.

This is illustrated in Julia as follows:

```
julia> p = chain(3) + chain(3);
julia> add_relation!(p, 4, 2);
julia> rem_vertex!(p, 2);
julia> collect(relations(p))
5-element Vector{Relation{Int64}}:
 Relation 1 < 3
 Relation 4 < 2
 Relation 4 < 3
```

```
 Relation 4 < 5
 Relation 5 < 2
```

Deleting Relations

Deleting a relation from a poset is complicated. The simplest case is the removal of a relation $a \prec b$ where b is a cover of a. In this case, it is possible just to remove the single relation $a \prec b$ and make no other changes to the poset. This is illustrated in Figure 2 in which we delete the cover relation $2 \prec 3$ from the linear order $1 \prec 2 \prec 3 \prec 4$.

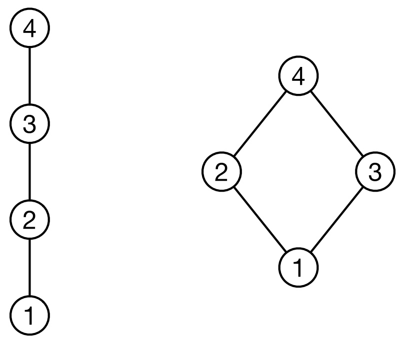

Figure 2: Deleting a cover relation, $2 \prec 3$, from a poset.

The following Julia code implements the action of deleting $2 \prec 3$ from a 4-element chain:

```
julia> p = chain(4);
julia> rem_relation!(p, 2, 3);
julia> collect(relations(p))
6-element Vector{Relation{Int64}}:
 Relation 1 < 2
 Relation 1 < 3
 Relation 1 < 4
 Relation 2 < 3
 Relation 2 < 4
 Relation 3 < 4
```

That this yields a partial order follows from Proposition 1.

The situation is different for non-cover relations. For example, consider the linear order $1 \prec 2 \prec 3 \prec 4 \prec 5$. Suppose we wish to delete the relation $2 \prec 4$. If we only delete that one relation, we would still have $2 \prec 3$ and $3 \prec 4$, so omitting $2 \prec 4$ would result in a violation of transitivity.

More generally, suppose we wish to remove the relation $a \prec b$ from a poset. If there is an element x with $a \prec x \prec b$ we cannot delete just $a \prec b$ and keep both $a \prec x$ and $x \prec b$.

There is no *a priori* reason to prefer one of $a \prec x$ or $x \prec b$ for deletion (or retention). Hence, it is a design decision that when removing a relation $a \prec b$ we also remove both relations $a \prec x$ and $x \prec b$ for all x between a and b. Proposition 1 ensures that the new relation gives a partial order.

For example, suppose we wish to delete the relation $2 \prec 4$ from the linear order $1 \prec 2 \prec 3 \prec 4 \prec 5$. Our implementation deletes not only $2 \prec 4$ but also $2 \prec 3$ and $3 \prec 4$ as well. This is illustrated in Figure 3.

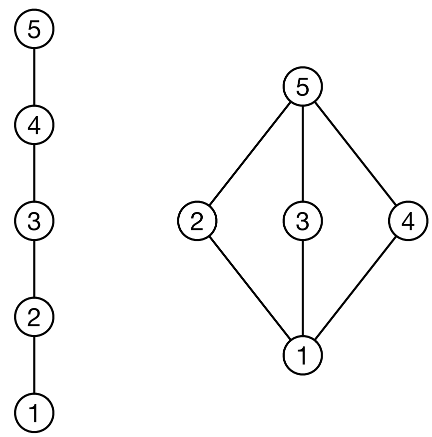

Figure 3: Deleting the non-cover relation $2 \prec 4$ from the poset on the left yields the poset on the right. The function `rem_relation!(p,a,b)` not only deletes $a \prec b$, but also all relations of the form $a \prec x$ and $x \prec b$ for vertices x between a and b.

This Julia code illustrates the deletion:

```
julia> p = chain(5);
julia> rem_relation!(p, 2, 4);
julia> collect(relations(p))
7-element Vector{Relation{Int64}}:
 Relation 1 < 2
 Relation 1 < 3
 Relation 1 < 4
 Relation 1 < 5
 Relation 2 < 5
 Relation 3 < 5
 Relation 4 < 5
```

Proposition 1. *Let $P = (V, \prec)$ be a poset. Let $a, b \in V$ with $a \prec b$. Let $\prec'$ be a new relation on V in which $x \prec' y$ provided $x \prec y$ and neither $x = a \prec y \preceq b$ nor $a \preceq x \prec y = b$.*

Then $(V, \prec')$ is a partially ordered set.

Proof. Note that if $x \prec' y$ then necessarily $x \prec y$. [Formally, $\prec' \subseteq \prec$.]

We cannot have $x \prec' x$ because that would imply $x \prec x$. Likewise, we cannot have $x \prec' y$ and $y \prec' x$ as that would imply $x \prec y$ and $y \prec x$. Hence $\prec'$ is irreflexive and antisymmetric. We need to prove that $\prec'$ is transitive.

Suppose $x \prec' y \prec' z$. We must show $x \prec' z$. Note that $x \prec' y \prec' z$ implies $x \prec y \prec z$ and hence $x \prec z$. To show that $x \prec' z$ we need to rule out both $x = a \prec z \preceq b$ and $a \preceq x \prec b = z$.

Suppose $x = a \prec z \preceq b$. Since y is between x and z, we have $x = a \prec y \prec z \preceq b$ implying that $x = a \not\prec' y$, a contradiction.

Supposing $a \preceq x \prec b = z$ leads to a similar contradiction.

Therefore $x \prec' z$ and we conclude that $\prec'$ is transitive and that $(V, \prec')$ is a partially ordered set. □

—Ed Scheinerman (`ers@jhu.edu`)