1 Breadth-First Search

```
STk> (begin (map (lambda (x) (bfs-result x)) und-graphs) #f)
{[ 1 2 3 4 5 6 7 ]{ { 1 2 }{ 1 3 }{ 1 4 }{ 1 5 }{ 1 6 }{ 1 7 }}}
< 1 2 3 4 5 6 7 >

{[ 1 2 3 4 5 6 7 ]{ { 1 2 }{ 1 3 }{ 1 4 }{ 1 5 }{ 1 6 }{ 1 7 }{ 2 3 }{ 2 7 }{ 3 4 }{ 4 5 }
{ 5 6 }
{ 6 7 }}}
< 1 2 3 4 5 6 7 >

{[ 1 2 3 4 5 6 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 3 4 }{ 4 5 }{ 4 6 }{ 5 6 }}}
< 1 2 3 4 5 6 >

{[ 0 1 2 3 4 ]{ { 0 1 }{ 0 4 }{ 1 2 }{ 2 3 }{ 3 4 }}}
< 0 1 4 2 3 >

{[ ]{ }}
< >

{[ 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 ]{
 { 0 1 }{ 0 5 }{ 1 2 }{ 1 6 }{ 2 3 }{ 2 7 }{ 3 4 }{ 3 8 }{ 4 9 }{ 5 6 }
{ 5 10 }{ 6 7 }{ 6 11 }{ 7 8 }{ 7 12 }{ 8 9 }{ 8 13 }{ 9 14 }{ 10 11 }{ 10 15 }
{ 11 12 }{ 11 16 }{ 12 13 }{ 12 17 }{ 13 14 }{ 13 18 }{ 14 19 }{ 15 16 }{ 15 20 }
{ 16 17 }{ 16 21 }{ 17 18 }{ 17 22 }{ 18 19 }{ 18 23 }{ 19 24 }{ 20 21 }
{ 21 22 }
{ 22 23 }{ 23 24 }}}
< 0 1 5 2 6 10 3 7 11 15 4 8 12 16 20 9 13 17 21 14 18 22 19 23 24 >

{[ 1 2 3 4 5 6 7 8 9 10 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 3 4 }{ 4 5 }{ 4 6 }{ 5 6 }}}
< 1 2 3 4 5 6 >

{[ 1 2 3 4 5 6 ]{ { 1 4 }{ 1 5 }{ 1 6 }{ 2 4 }{ 2 5 }{ 2 6 }{ 3 4 }{ 3 5 }{ 3 6 }}}
< 1 4 5 6 2 3 >

{[ 0 1 2 3 4 ]{ { 0 1 }{ 0 2 }{ 0 3 }{ 0 4 }{ 1 2 }{ 1 3 }{ 1 4 }{ 2 3 }{ 2 4 }{ 3 4 }}}
< 0 1 2 3 4 >

{[ 1 2 3 4 ]{ { 1 2 }{ 1 2 }{ 1 3 }{ 1 3 }{ 2 4 }{ 3 4 }}}
< 1 2 3 4 >

{[ 1 2 3 4 5 6 7 8 9 10 ]{ { 1 2 }{ 1 3 }{ 1 6 }{ 2 4 }{ 2 7 }{ 3 5 }{ 3 8 }{ 4 5 }{ 4 9 }{ 5 10 }
{ 6 9 }{ 6 10 }{ 7 8 }{ 7 10 }{ 8 9 }}}
```

```
< 1 2 3 6 4 7 5 8 9 10 >

{[ 1 2 3 4 5 6 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 4 5 }{ 4 6 }{ 5 6 }}}
< 1 2 3 >

#f

STk> (begin (map (lambda (x) (bfs-result x)) dir-graphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 >< 2 3 >< 3 4 >< 4 5 >< 5 1 >}}
< 1 2 3 4 5 >

{[ 1 2 3 4 5 6 7 8 ]{ < 1 2 >< 1 3 >< 1 4 >< 2 3 >< 3 5 >< 3 7 >< 4 6 >< 5 8 >< 6 8 >< 7 6 >}}
< 1 2 3 4 5 7 6 8 >

{[ 1 2 3 4 5 ]{ < 1 2 >< 1 3 >< 2 3 >< 2 4 >< 3 4 >< 3 5 >< 4 1 >< 4 5 >< 5 1 >< 5 2 >}}
< 1 2 3 4 5 >

{[ 1 2 3 4 5 ]{ < 1 2 >< 1 2 >< 1 3 >< 1 3 >< 1 4 >< 3 1 >}}
< 1 2 3 4 >

{[ 1 ]{ < 1 1 >}}
< 1 >

{[ 1 2 3 4 5 6 7 8 ]{ < 1 2 >< 1 3 >< 1 4 >< 1 7 >< 2 5 >< 2 8 >< 3 6 >< 3 7 >< 4 7 >< 5 1 >}}
< 1 2 3 4 7 5 8 6 >

{[ 1 2 3 4 5 6 ]{ < 1 2 >< 2 3 >< 3 1 >< 4 2 >< 4 6 >< 5 4 >< 6 5 >}}
< 1 2 3 >

#f

STk> (begin (map (lambda (x) (bfs-result x)) mixed-graphs) #f)
{[ 1 2 3 ]{ < 1 3 >< 2 1 >{ 2 3 }}}
< 1 3 2 >

#f

STk> (begin (map (lambda (x) (bfs-result x)) und-hypergraphs) #f)
{[ 1 2 3 4 5 6 7 8 9 ]{ { 1 2 3 }{ 1 4 }{ 2 3 5 }{ 4 5 6 }{ 7 8 9 }}}
< 1 3 2 4 5 6 >

{[ 1 2 3 4 5 6 ]{ { 1 2 3 }{ 1 4 }{ 2 3 5 }{ 4 5 6 }}}
< 1 3 2 4 5 6 >

{[ 1 2 3 4 5 ]{ { 1 2 3 }{ 1 2 3 }{ 2 4 }{ 3 5 }{ 4 5 }}}
```

```
< 1 3 2 5 4 >

{[ 1 2 3 ]{ { 1 2 3 }}}
< 1 3 2 >

#f

STk> (begin (map (lambda (x) (bfs-result x)) dir-hypergraphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 3 >< 4 2 3 >}}
< 1 3 2 >

{[ 1 2 3 ]{ < 1 2 3 >< 1 2 3 >}}
< 1 3 2 >

#f

STk> (begin (map (lambda (x) (bfs-result x)) mixed-hypergraphs) #f)
{[ 1 2 3 ]{ < 1 2 3 >{ 2 3 }}}
< 1 3 2 >

#f
```

2 Depth-First Search

```
STk> (begin (map (lambda (x) (dfs-result x)) und-graphs) #f)
{[ 1 2 3 4 5 6 7 ]{ { 1 2 }{ 1 3 }{ 1 4 }{ 1 5 }{ 1 6 }{ 1 7 }}}
< 1 2 3 4 5 6 7 >

{[ 1 2 3 4 5 6 7 ]{ { 1 2 }{ 1 3 }{ 1 4 }{ 1 5 }{ 1 6 }{ 1 7 }{ 2 3 }{ 2 7 }{ 3 4 }{ 4 5 }
{ 5 6 }
{ 6 7 }}}
< 1 2 3 4 5 6 7 >

{[ 1 2 3 4 5 6 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 3 4 }{ 4 5 }{ 4 6 }{ 5 6 }}}
< 1 2 3 4 5 6 >

{[ 0 1 2 3 4 ]{ { 0 1 }{ 0 4 }{ 1 2 }{ 2 3 }{ 3 4 }}}
< 0 1 2 3 4 >

{[ ]{ }}
< >
```

```
{[ 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 ]{
 { 0 1 }{ 0 5 }{ 1 2 }{ 1 6 }{ 2 3 }{ 2 7 }{ 3 4 }{ 3 8 }{ 4 9 }{ 5 6 }
{ 5 10 }{ 6 7 }{ 6 11 }{ 7 8 }{ 7 12 }{ 8 9 }{ 8 13 }{ 9 14 }{ 10 11 }{ 10 15 }
{ 11 12 }{ 11 16 }{ 12 13 }{ 12 17 }{ 13 14 }{ 13 18 }{ 14 19 }{ 15 16 }{ 15 20 }
{ 16 17 }{ 16 21 }{ 17 18 }{ 17 22 }{ 18 19 }{ 18 23 }{ 19 24 }{ 20 21 }
{ 21 22 }
{ 22 23 }{ 23 24 }}}
< 0 1 2 3 4 9 8 7 6 5 10 11 12 13 14 19 18 17 16 15 20 21 22 23 24 >

{[ 1 2 3 4 5 6 7 8 9 10 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 3 4 }{ 4 5 }{ 4 6 }{ 5 6 }}}
< 1 2 3 4 5 6 7 8 9 10 >

{[ 1 2 3 4 5 6 ]{ { 1 4 }{ 1 5 }{ 1 6 }{ 2 4 }{ 2 5 }{ 2 6 }{ 3 4 }{ 3 5 }{ 3 6 }}}
< 1 4 2 5 3 6 >

{[ 0 1 2 3 4 ]{ { 0 1 }{ 0 2 }{ 0 3 }{ 0 4 }{ 1 2 }{ 1 3 }{ 1 4 }{ 2 3 }{ 2 4 }{ 3 4 }}}
< 0 1 2 3 4 >

{[ 1 2 3 4 ]{ { 1 2 }{ 1 2 }{ 1 3 }{ 1 3 }{ 2 4 }{ 3 4 }}}
< 1 2 4 3 >

{[ 1 2 3 4 5 6 7 8 9 10 ]{ { 1 2 }{ 1 3 }{ 1 6 }{ 2 4 }{ 2 7 }{ 3 5 }{ 3 8 }{ 4 5 }{ 4 9 }{ 5 10 }
{ 6 9 }{ 6 10 }{ 7 8 }{ 7 10 }{ 8 9 }}}
< 1 2 4 5 3 8 7 10 6 9 >

{[ 1 2 3 4 5 6 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 4 5 }{ 4 6 }{ 5 6 }}}
< 1 2 3 4 5 6 >

#f

STk> (begin (map (lambda (x) (dfs-result x)) dir-graphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 >< 2 3 >< 3 4 >< 4 5 >< 5 1 >}}
< 1 2 3 4 5 >

{[ 1 2 3 4 5 6 7 8 ]{ < 1 2 >< 1 3 >< 1 4 >< 2 3 >< 3 5 >< 3 7 >< 4 6 >< 5 8 >< 6 8 >< 7 6 >}}
< 1 2 3 5 8 7 6 4 >

{[ 1 2 3 4 5 ]{ < 1 2 >< 1 3 >< 2 3 >< 2 4 >< 3 4 >< 3 5 >< 4 1 >< 4 5 >< 5 1 >< 5 2 >}}
< 1 2 3 4 5 >

{[ 1 2 3 4 5 ]{ < 1 2 >< 1 2 >< 1 3 >< 1 3 >< 1 4 >< 3 1 >}}
< 1 2 3 4 5 >

{[ 1 ]{ < 1 1 >}}
```

```
< 1 >

{[ 1 2 3 4 5 6 7 8 ]{ < 1 2 >< 1 3 >< 1 4 >< 1 7 >< 2 5 >< 2 8 >< 3 6 >< 3 7 >< 4 7 >< 5 1 >}}
< 1 2 5 8 3 6 7 4 >

{[ 1 2 3 4 5 6 ]{ < 1 2 >< 2 3 >< 3 1 >< 4 2 >< 4 6 >< 5 4 >< 6 5 >}}
< 1 2 3 4 6 5 >

#f

STk> (begin (map (lambda (x) (dfs-result x)) mixed-graphs) #f)
{[ 1 2 3 ]{ < 1 3 >< 2 1 >{ 2 3 }}}
< 1 3 2 >

#f

STk> (begin (map (lambda (x) (dfs-result x)) und-hypergraphs) #f)
{[ 1 2 3 4 5 6 7 8 9 ]{ { 1 2 3 }{ 1 4 }{ 2 3 5 }{ 4 5 6 }{ 7 8 9 }}}
< 1 3 5 6 4 2 7 9 8 >

{[ 1 2 3 4 5 6 ]{ { 1 2 3 }{ 1 4 }{ 2 3 5 }{ 4 5 6 }}}
< 1 3 5 6 4 2 >

{[ 1 2 3 4 5 ]{ { 1 2 3 }{ 1 2 3 }{ 2 4 }{ 3 5 }{ 4 5 }}}
< 1 3 2 4 5 >

{[ 1 2 3 ]{ { 1 2 3 }}}
< 1 3 2 >

#f

STk> (begin (map (lambda (x) (dfs-result x)) dir-hypergraphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 3 >< 4 2 3 >}}
< 1 3 2 4 5 >

{[ 1 2 3 ]{ < 1 2 3 >< 1 2 3 >}}
< 1 3 2 >

#f

STk> (begin (map (lambda (x) (dfs-result x)) mixed-hypergraphs) #f)
{[ 1 2 3 ]{ < 1 2 3 >{ 2 3 }}}
< 1 3 2 >

#f
```

---

3 Strongly Connected Components

```
STk> (begin (map (lambda (x) (scc-result x)) und-graphs) #f)
{[ 1 2 3 4 5 6 7 ]{ { 1 2 }{ 1 3 }{ 1 4 }{ 1 5 }{ 1 6 }{ 1 7 }}}
Error: StronglyConnectedComponents (warning): Graph should be directed binary

*** Error:
    No applicable method for #[<generic> #p63e518]
in call (collection-vertex->string #[<set<set<vertex*>>> #p66a81c])
Current eval stack:
__________________
  0    (error "No applicable method for S̃
nin call S̃" gf (append (cons (slot-ref gf (quote name)) args)))
  1    (collection-vertex->string (scc g))
  2    (display (collection-vertex->string (scc g)))
  3    (scc-result x)
  4    (map (lambda (x) (scc-result x)) und-graphs)
  5    (begin (map (lambda (x) (scc-result x)) und-graphs) #f)

STk> (begin (map (lambda (x) (scc-result x)) dir-graphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 >< 2 3 >< 3 4 >< 4 5 >< 5 1 >}}

*** Error:
    No applicable method for #[<generic> #p63e518]
in call (collection-vertex->string #[<set<set<vertex*>>> #p63ea88])
Current eval stack:
__________________
  0    (error "No applicable method for S̃
nin call S̃" gf (append (cons (slot-ref gf (quote name)) args)))
  1    (collection-vertex->string (scc g))
  2    (display (collection-vertex->string (scc g)))
  3    (scc-result x)
  4    (map (lambda (x) (scc-result x)) dir-graphs)
  5    (begin (map (lambda (x) (scc-result x)) dir-graphs) #f)

STk> (begin (map (lambda (x) (scc-result x)) mixed-graphs) #f)
{[ 1 2 3 ]{ < 1 3 >< 2 1 >{ 2 3 }}}
Error: StronglyConnectedComponents (warning): Graph should be directed binary

*** Error:
    No applicable method for #[<generic> #p63e518]
in call (collection-vertex->string #[<set<set<vertex*>>> #p61f150])
Current eval stack:
__________________
```

```
  0    (error "No applicable method for S̃
nin call S̃" gf (append (cons (slot-ref gf (quote name)) args)))
  1    (collection-vertex->string (scc g))
  2    (display (collection-vertex->string (scc g)))
  3    (scc-result x)
  4    (map (lambda (x) (scc-result x)) mixed-graphs)
  5    (begin (map (lambda (x) (scc-result x)) mixed-graphs) #f)

STk> (begin (map (lambda (x) (scc-result x)) und-hypergraphs) #f)
{[ 1 2 3 4 5 6 7 8 9 ]{ { 1 2 3 }{ 1 4 }{ 2 3 5 }{ 4 5 6 }{ 7 8 9 }}}
Error: StronglyConnectedComponents (warning): Graph should be directed binary

*** Error:
    No applicable method for #[<generic> #p63e518]
in call (collection-vertex->string #[<set<set<vertex*>>> #p50a6c4])
Current eval stack:
__________________
  0    (error "No applicable method for S̃
nin call S̃" gf (append (cons (slot-ref gf (quote name)) args)))
  1    (collection-vertex->string (scc g))
  2    (display (collection-vertex->string (scc g)))
  3    (scc-result x)
  4    (map (lambda (x) (scc-result x)) und-hypergraphs)
  5    (begin (map (lambda (x) (scc-result x)) und-hypergraphs) #f)

STk> (begin (map (lambda (x) (scc-result x)) dir-hypergraphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 3 >< 4 2 3 >}}
Error: StronglyConnectedComponents (warning): Graph should be directed binary

*** Error:
    No applicable method for #[<generic> #p63e518]
in call (collection-vertex->string #[<set<set<vertex*>>> #p65d52c])
Current eval stack:
__________________
  0    (error "No applicable method for S̃
nin call S̃" gf (append (cons (slot-ref gf (quote name)) args)))
  1    (collection-vertex->string (scc g))
  2    (display (collection-vertex->string (scc g)))
  3    (scc-result x)
  4    (map (lambda (x) (scc-result x)) dir-hypergraphs)
  5    (begin (map (lambda (x) (scc-result x)) dir-hypergraphs) #f)

STk> (begin (map (lambda (x) (scc-result x)) mixed-hypergraphs) #f)
{[ 1 2 3 ]{ < 1 2 3 >{ 2 3 }}}
Error: StronglyConnectedComponents (warning): Graph should be directed binary
```

```
*** Error:
    No applicable method for #[<generic> #p63e518]
in call (collection-vertex->string #[<set<set<vertex*>>> #p64609c])
Current eval stack:
__________________
  0    (error "No applicable method for S̃
nin call S̃" gf (append (cons (slot-ref gf (quote name)) args)))
  1    (collection-vertex->string (scc g))
  2    (display (collection-vertex->string (scc g)))
  3    (scc-result x)
  4    (map (lambda (x) (scc-result x)) mixed-hypergraphs)
  5    (begin (map (lambda (x) (scc-result x)) mixed-hypergraphs) #f)
```

4 Topological Sort

```
STk> (begin (map (lambda (x) (topsort-result x)) und-graphs) #f)
{[ 1 2 3 4 5 6 7 ]{ { 1 2 }{ 1 3 }{ 1 4 }{ 1 5 }{ 1 6 }{ 1 7 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 4 5 6 7 ]{ { 1 2 }{ 1 3 }{ 1 4 }{ 1 5 }{ 1 6 }{ 1 7 }{ 2 3 }{ 2 7 }{ 3 4 }{ 4 5 }
{ 5 6 }
{ 6 7 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 4 5 6 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 3 4 }{ 4 5 }{ 4 6 }{ 5 6 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 0 1 2 3 4 ]{ { 0 1 }{ 0 4 }{ 1 2 }{ 2 3 }{ 3 4 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ ]{ }}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 ]{
 { 0 1 }{ 0 5 }{ 1 2 }{ 1 6 }{ 2 3 }{ 2 7 }{ 3 4 }{ 3 8 }{ 4 9 }{ 5 6 }
{ 5 10 }{ 6 7 }{ 6 11 }{ 7 8 }{ 7 12 }{ 8 9 }{ 8 13 }{ 9 14 }{ 10 11 }{ 10 15 }
```

```
{ 11 12 }{ 11 16 }{ 12 13 }{ 12 17 }{ 13 14 }{ 13 18 }{ 14 19 }{ 15 16 }{ 15 20 }
{ 16 17 }{ 16 21 }{ 17 18 }{ 17 22 }{ 18 19 }{ 18 23 }{ 19 24 }{ 20 21 }
{ 21 22 }
{ 22 23 }{ 23 24 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 4 5 6 7 8 9 10 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 3 4 }{ 4 5 }{ 4 6 }{ 5 6 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 4 5 6 ]{ { 1 4 }{ 1 5 }{ 1 6 }{ 2 4 }{ 2 5 }{ 2 6 }{ 3 4 }{ 3 5 }{ 3 6 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 0 1 2 3 4 ]{ { 0 1 }{ 0 2 }{ 0 3 }{ 0 4 }{ 1 2 }{ 1 3 }{ 1 4 }{ 2 3 }{ 2 4 }{ 3 4 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 4 ]{ { 1 2 }{ 1 2 }{ 1 3 }{ 1 3 }{ 2 4 }{ 3 4 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 4 5 6 7 8 9 10 ]{ { 1 2 }{ 1 3 }{ 1 6 }{ 2 4 }{ 2 7 }{ 3 5 }{ 3 8 }{ 4 5 }{ 4 9 }{ 5 10 }
{ 6 9 }{ 6 10 }{ 7 8 }{ 7 10 }{ 8 9 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 4 5 6 ]{ { 1 2 }{ 1 3 }{ 2 3 }{ 4 5 }{ 4 6 }{ 5 6 }}}
Error: TopologicalSort: Graph must be directed binary
< >

#f

STk> (begin (map (lambda (x) (topsort-result x)) dir-graphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 >< 2 3 >< 3 4 >< 4 5 >< 5 1 >}}
Error: TopologicalSort: Graph must be acyclic
< >

{[ 1 2 3 4 5 6 7 8 ]{ < 1 2 >< 1 3 >< 1 4 >< 2 3 >< 3 5 >< 3 7 >< 4 6 >< 5 8 >< 6 8 >< 7 6 >}}
< 1 4 2 3 7 6 5 8 >

{[ 1 2 3 4 5 ]{ < 1 2 >< 1 3 >< 2 3 >< 2 4 >< 3 4 >< 3 5 >< 4 1 >< 4 5 >< 5 1 >< 5 2 >}}
Error: TopologicalSort: Graph must be acyclic
```

< >

{[1 2 3 4 5]{ < 1 2 >< 1 2 >< 1 3 >< 1 3 >< 1 4 >< 3 1 >}}
Error: TopologicalSort: Graph must be acyclic
< >

{[1]{ < 1 1 >}}
< 1 >

{[1 2 3 4 5 6 7 8]{ < 1 2 >< 1 3 >< 1 4 >< 1 7 >< 2 5 >< 2 8 >< 3 6 >< 3 7 >< 4 7 >< 5 1 >}}
Error: TopologicalSort: Graph must be acyclic
< >

{[1 2 3 4 5 6]{ < 1 2 >< 2 3 >< 3 1 >< 4 2 >< 4 6 >< 5 4 >< 6 5 >}}
Error: TopologicalSort: Graph must be acyclic
< >

#f

```
STk> (begin (map (lambda (x) (topsort-result x)) mixed-graphs) #f)
```
{[1 2 3]{ < 1 3 >< 2 1 >{ 2 3 }}}
Error: TopologicalSort: Graph must be directed binary
< >

#f

```
STk> (begin (map (lambda (x) (topsort-result x)) und-hypergraphs) #f)
```
{[1 2 3 4 5 6 7 8 9]{ { 1 2 3 }{ 1 4 }{ 2 3 5 }{ 4 5 6 }{ 7 8 9 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[1 2 3 4 5 6]{ { 1 2 3 }{ 1 4 }{ 2 3 5 }{ 4 5 6 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[1 2 3 4 5]{ { 1 2 3 }{ 1 2 3 }{ 2 4 }{ 3 5 }{ 4 5 }}}
Error: TopologicalSort: Graph must be directed binary
< >

{[1 2 3]{ { 1 2 3 }}}
Error: TopologicalSort: Graph must be directed binary
< >

#f

```
STk> (begin (map (lambda (x) (topsort-result x)) dir-hypergraphs) #f)
{[ 1 2 3 4 5 ]{ < 1 2 3 >< 4 2 3 >}}
Error: TopologicalSort: Graph must be directed binary
< >

{[ 1 2 3 ]{ < 1 2 3 >< 1 2 3 >}}
Error: TopologicalSort: Graph must be directed binary
< >

#f

STk> (begin (map (lambda (x) (topsort-result x)) mixed-hypergraphs) #f)
{[ 1 2 3 ]{ < 1 2 3 >{ 2 3 }}}
Error: TopologicalSort: Graph must be directed binary
< >

#f
```