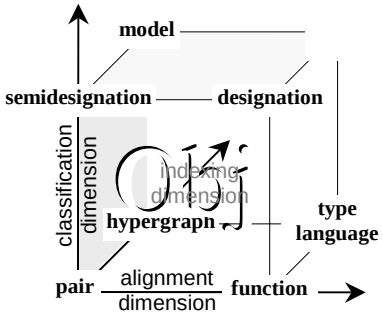


The Namespace of Hypergraphs

Table 1 lists the terminology in the namespace of hypergraphs.

Table 1: The Terminology in the Namespace of Hypergraphs

	Class	Function	Other
hgph	hypergraph	reference tuple edge-arity reference-arity name node edge	
		arity case injection indication projection arity-morphism case-tuple comediator spangraph	
hgph .mor	hypergraph-morphism	source target reference tuple edge-arity reference-arity name node edge composition identity eta	composable-opspan composable
		arity-morphism case indication projection comediator spangraph-morphism	

Hypergraphs

hgph

This section discusses hypergraphs. First, we give a concise mathematical definition (Figure 1), and then we discuss and formalize the various parts of this definition.

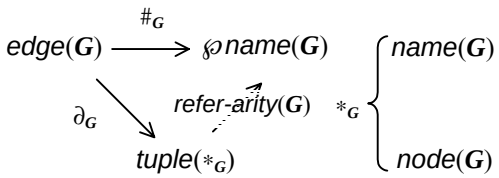


Figure 1: Hypergraph

- A *hypergraph* $\boldsymbol{G} = \langle refer(\boldsymbol{G}), tuple(\boldsymbol{G})\rangle$ consists of
  - a *reference* set pair $*_{\boldsymbol{G}} = refer(\boldsymbol{G})$
  - a *tuple* function $\partial_{\boldsymbol{G}} = tuple(\boldsymbol{G})$

  that satisfy the pullback constraint

  $tgt(tuple(\boldsymbol{G})) = tuple(refer(\boldsymbol{G}))$.

```
(1) (SET$class hypergraph)

(2) (SET.FTN$function reference)
    (= (SET.FTN$source reference) hypergraph)
    (= (SET.FTN$target reference) set.pr$pair)

(3) (SET.FTN$function tuple)
    (= (SET.FTN$source tuple) hypergraph)
    (= (SET.FTN$target tuple) set.ftn$function)

(4) (= (SET.FTN$composition [tuple set.ftn$target])
       (SET.FTN$composition [reference set.pr$tuple]))
```

- For convenience of theoretical presentation, we introduce additional hypergraph terminology for the composition between the reference function and the tuple arity function.
  - *reference-arity* function $refer\text{-}arity(\boldsymbol{G}) = tuple\text{-}arity(*_{\boldsymbol{G}}) : tuple(refer(\boldsymbol{G})) \to \wp name(\boldsymbol{G})$.

```
(5) (SET.FTN$function reference-arity)
    (= (SET.FTN$source reference-arity) hypergraph)
    (= (SET.FTN$target reference-arity) set.ftn$function)
    (= reference-arity (SET.FTN$composition [reference set.pr$tuple-arity]))
```

- The set function composition of the tuple function with the reference arity function defines
  - an *edge arity* function $\#_{\boldsymbol{G}} = edge\text{-}arity(\boldsymbol{G}) = tuple(\boldsymbol{G}) \cdot refer\text{-}arity(\boldsymbol{G})$.

```
(6) (SET.FTN$function edge-arity)
    (= (SET.FTN$source edge-arity) hypergraph)
    (= (SET.FTN$target edge-arity) set.ftn$function)
    (forall (?g (hypergraph ?g))
         (= (edge-arity ?g)
            (set.ftn$composition [(tuple ?g) (reference-arity ?g)])))
```

- For convenience of practical reference, we introduce additional hypergraph terminology for the source and target components of these functions. The source of the tuple function is called the *edge* set of $\boldsymbol{G}$ and denoted $edge(\boldsymbol{G})$. The second of the reference pair is called the *node* set of $\boldsymbol{G}$ and denoted $node(\boldsymbol{G})$. The first of the reference pair is called the *name* set of $\boldsymbol{G}$ and denoted $name(\boldsymbol{G})$. In summary, we provide terminology for the following sets:
  - the *node* set $node(\boldsymbol{G}) = set2(refer(\boldsymbol{G}))$,
  - the *edge* set $edge(\boldsymbol{G}) = src(tuple(\boldsymbol{G}))$, and
  - the *name* set $name(\boldsymbol{G}) = set1(refer(\boldsymbol{G}))$.

  This results in the following presentations of the signature, arity and reference functions:
  - *tuple* function $\partial_{\boldsymbol{G}} = tuple(\boldsymbol{G}) : edge(\boldsymbol{G}) \to tuple(refer(\boldsymbol{G}))$,
  - *edge-arity* function $\#_{\boldsymbol{G}} = edge\text{-}arity(\boldsymbol{G}) : edge(\boldsymbol{G}) \to \wp name(\boldsymbol{G})$, and
  - *reference* pair $refer(\boldsymbol{G}) = \langle name(\boldsymbol{G}), node(\boldsymbol{G})\rangle$.

  Names refer to nodes – a tuple in $tuple(refer(\boldsymbol{G}))$ maps a name to its referent node. We assume that each hypergraph has an adequate, possibly denumerable, set of names $name(\boldsymbol{G})$. This generalizes the

usual case where sequences are used – the advantage for this generality is elimination of the dependency on sequences. One way to return to sequences is to define $\mathit{name}(\boldsymbol{G}) = \mathit{index} \times \mathit{node}(\boldsymbol{G})$ for some index set such as $\mathit{index} = \mathit{Natno}$.

The hyperedges in a hypergraph are many-sorted, which means that the component nodes in tuples are named. For any hyperedge $\varepsilon \in \mathit{edge}(\boldsymbol{G})$ the *tuple* of ε is a function $\partial_{\boldsymbol{G}}(\varepsilon) : \mathit{arity}(\boldsymbol{G})(\varepsilon) \to \mathit{node}(\boldsymbol{G})$. The set of names $\mathit{arity}(\boldsymbol{G})(\varepsilon)$ that reference the nodes in the tuple of a hyperedge $\varepsilon \in \mathit{edge}(\boldsymbol{G})$ is called its *arity*. The arity of hyperedges can overlap. Hence, $\partial_{\boldsymbol{G}}(\varepsilon)$ is a tuple of nodes with indexing set $\mathit{arity}(\boldsymbol{G})(\varepsilon) \subseteq \mathit{name}(\boldsymbol{G})$.

An external arity form for a hyperedge is $(\varepsilon, x_1, x_2, \ldots x_n)$, where $\mathit{arity}(\boldsymbol{G})(\varepsilon) = \{x_1, x_2, \ldots x_n)$. An external signature form for a hyperedge is $(\varepsilon, x_1 : \nu_1, x_2 : \nu_2, \ldots x_n : \nu_n)$, where $\partial_{\boldsymbol{G}}(\varepsilon) = (x_1 : \nu_1, x_2 : \nu_2, \ldots x_n : \nu_n)$.

```
(7) (SET.FTN$function node)
    (= (SET.FTN$source node) hypergraph)
    (= (SET.FTN$target node) set$set)
    (= node (SET.FTN$composition [reference set.pr$set2]))

(8) (SET.FTN$function edge)
    (= (SET.FTN$source edge) hypergraph)
    (= (SET.FTN$target edge) set$set)
    (= edge (SET.FTN$composition [tuple set.ftn$source]))

(9) (SET.FTN$function name)
    (= (SET.FTN$source name) hypergraph)
    (= (SET.FTN$target name) set$set)
    (= name (SET.FTN$composition [reference set.pr$set1]))
```

Case and Spangraphs

- Any hypergraph $\boldsymbol{G} = \langle refer(\boldsymbol{G}), tuple(\boldsymbol{G})\rangle$ has a coproduct *arity*

  $\#_{\boldsymbol{G}} = arity(\boldsymbol{G}) : edge(\boldsymbol{G}) \to \wp name(\boldsymbol{G})$.

```
(10) (SET.FTN$function arity)
     (= (SET.FTN$source arity) hypergraph)
     (= (SET.FTN$target arity) set.col.art$arity)
     (= (SET.FTN$composition [arity set.col.art$index]) edge)
     (= (SET.FTN$composition [arity set.col.art$base]) name)
     (= (SET.FTN$composition [arity set.col.art$function]) edge-arity)
```

- Any hypergraph $\boldsymbol{G}$ has a set of *cases*

  $case(\boldsymbol{G}) = \sum arity(\boldsymbol{G})$

  $= \sum_{\varepsilon \in edge(\boldsymbol{G})} arity(\boldsymbol{G})(\varepsilon)$

  $= \{(\varepsilon, x) \mid \varepsilon \in edge(\boldsymbol{G}), x \in arity(\boldsymbol{G})(\varepsilon)\}$,

  that is the coproduct of its arity.

  For any hypergraph $\boldsymbol{G}$ and any hyperedge $\varepsilon \in edge(\boldsymbol{G})$ there is a case *injection* function:

  $inj(\boldsymbol{G})(\varepsilon) : \#_{\boldsymbol{G}}(\varepsilon) = arity_{\boldsymbol{G}}(\varepsilon) \to case(\boldsymbol{G})$

  defined by $inj(\boldsymbol{G})(\varepsilon)(x) = (\varepsilon, x)$ for all hyperedges $\varepsilon \in edge(\boldsymbol{G})$ and all names $x \in arity(\boldsymbol{G})(\varepsilon)$. Obviously, the injections are injective. They commute (Diagram 1) with projection and inclusion.

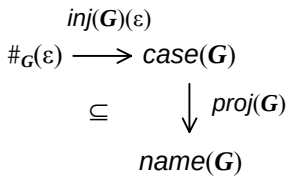


Diagram 1: Coproduct

```
(11) (SET.FTN$function case)
     (= (SET.FTN$source case) hypergraph)
     (= (SET.FTN$target case) set$set)
     (= case (SET.FTN$composition [arity set.col.art$coproduct]))

(12) (KIF$function injection)
     (= (KIF$source injection) hypergraph)
     (= (KIF$target injection) SET.FTN$function)
     (= injection (SET.FTN$composition [arity set.col.art$injection]))
```

- Any hypergraph $\boldsymbol{G}$ defines *indication* and *projection* functions (Figure 2) based on its arity:

  $indic(\boldsymbol{G}) : case(\boldsymbol{G}) \to edge(\boldsymbol{G})$,

  $proj(\boldsymbol{G}) : case(\boldsymbol{G}) \to name(\boldsymbol{G})$.

  These are defined by

  $indic(\boldsymbol{G})((\varepsilon, x)) = \varepsilon$ and $proj(\boldsymbol{G})((\varepsilon, x)) = x$

  for all hyperedges $\varepsilon \in edge(\boldsymbol{G})$ and all names $x \in arity(\boldsymbol{G})(\varepsilon)$.

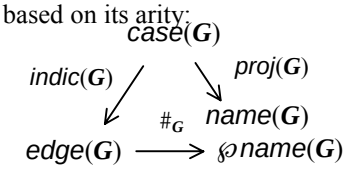


Figure 2: Indication and projection

```
(13) (SET.FTN$function indication)
     (= (SET.FTN$source indication) hypergraph)
     (= (SET.FTN$target indication) set.ftn$function)
     (= indication (SET.FTN$composition [arity set.col.art$indication]))

(14) (SET.FTN$function projection)
     (= (SET.FTN$source projection) hypergraph)
     (= (SET.FTN$target projection) set.ftn$function)
     (= projection (SET.FTN$composition [arity set.col.art$projection]))
```

- Any hypergraph $\boldsymbol{G}$ defines a morphism from its arity to the arity of its reference pair:

  $arity\text{-}mor(\boldsymbol{G}) = \langle tuple(\boldsymbol{G}), id\rangle : arity(\boldsymbol{G}) \to arity(refer(\boldsymbol{G}))$.

  This arity morphism will be used to define a case function between the case set axiomatized here and the case set of the reference pair. Since both of these are coproducts of arities, this case function will be define as the coproduct of the arity morphism. This case function is used to define comediation.

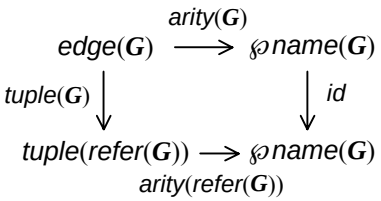


Figure 3: Arity Morphism

```
(15) (SET.FTN$function arity-morphism)
```

```
(= (SET.FTN$source arity-morphism) hypergaph)
(= (SET.FTN$target arity-morphism)
   set.col.art.mor$arity-morphism)
(= (SET.FTN$composition [arity-morphism set.col.art.mor$source]) arity)
(= (SET.FTN$composition [arity-morphism set.col.art.mor$target])
   (SET.FTN$composition [refer set.pr$arity]))
(= (SET.FTN$composition [arity-morphism set.col.art.mor$index]) tuple)
(= (SET.FTN$composition [arity-morphism set.col.art.mor$base])
   (SET.FTN$composition [(SET.FTN$composition [name set$power]) set.ftn$identity]))
```

- Any hypergraph $\boldsymbol{G}$ has a function (Figure 4)

  $\mathit{case\text{-}tuple}(\boldsymbol{G}) : \mathit{case}(\boldsymbol{G}) \to \mathit{case}(\mathit{refer}(\boldsymbol{G}))$,

  that satisfies the following constraints (Figure 2):

  $\mathit{case\text{-}tuple}(\boldsymbol{G}) \cdot \mathit{index}(\mathit{refer}(\boldsymbol{G})) = \mathit{index}(\boldsymbol{G}) \cdot \mathit{tuple}(\boldsymbol{G})$,

  $\mathit{case\text{-}tuple}\ (\boldsymbol{G}) \cdot \mathit{proj}(\mathit{refer}(\boldsymbol{G})) = \mathit{proj}(\boldsymbol{G})$.

  Pointwise, this is defined by

  $\mathit{case\text{-}tuple}(\boldsymbol{G})((\varepsilon, x)) = (\partial_{\boldsymbol{G}}(\varepsilon), x)$

  for all hyperedges $\varepsilon \in \mathit{edge}(\boldsymbol{G})$ and all names $x \in \mathit{arity}(\boldsymbol{G})(\varepsilon)$. Abstractly, this is defined to be the coproduct of the arity morphism of $\boldsymbol{G}$.

  It is the mediating function for the tuple function $\partial_{\boldsymbol{G}} = \mathit{tuple}(\boldsymbol{G}) : \mathit{edge}(\boldsymbol{G}) \to \mathit{tuple}(\mathit{refer}(\boldsymbol{G}))$, regarded as a coproduct cocone over the reference pair coproduct arity.

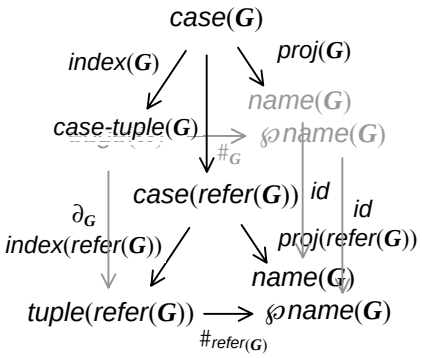


Figure 4: Case-tuple function

```
(16) (SET.FTN$function case-tuple)
     (= (SET.FTN$source case-tuple) hypergraph)
     (= (SET.FTN$target case-tuple) set.ftn$function)
     (= (SET.FTN$composition [case-tuple set.ftn$source]) case)
     (= (SET.FTN$composition [case-tuple set.ftn$target])
        (SET.FTN$composition [reference set.pr$case]))
     (= case-tuple
        (SET.FTN$composition [arity-morphism set.col.art.mor$coproduct]))
```

- Any hypergraph $\boldsymbol{G}$ defines a *comediator* function:

  $\tilde{*}_{\boldsymbol{G}} = \mathit{comed}(\boldsymbol{G}) : \mathit{case}(\boldsymbol{G}) \to \mathit{node}(\boldsymbol{G})$.

  This function is the slot-filler function for frames. Pointwise, it is defined by

  $\mathit{comed}(\boldsymbol{G})((\varepsilon, x)) = \partial_{\boldsymbol{G}}(\varepsilon)(x)$

  for all hyperedges $\varepsilon \in \mathit{edge}(\boldsymbol{G})$ and all names $x \in \mathit{arity}(\boldsymbol{G})(\varepsilon)$. Abstractly, it is the comediator – hence the name – of the coproduct cotuple (cocone) consisting of the collection of edge signatures regarded as functions,

  $\{\partial_{\boldsymbol{G}}(\varepsilon) : \mathit{arity}(\boldsymbol{G})(\varepsilon) \to \mathit{node}(\boldsymbol{G}) \mid \varepsilon \in \mathit{edge}(\boldsymbol{G})\}$.

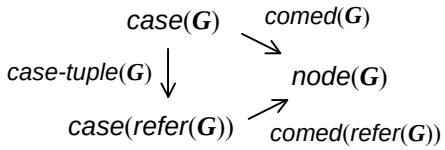


Diagram 2: Comediator

However, a simpler definition is at hand: the hypergraph comediator can be defined (Diagram 2) as the composition of the case-tuple function and the reference pair comediator. The comediator function commutes with the injection and tuple functions of any hyperedge. If the indexed names in $\mathit{case}(\boldsymbol{G})$ are viewed as roles (prehensions), the comediator is a reference function from roles to objects (actualities).

```
(17) (SET.FTN$function comediator)
     (= (SET.FTN$source comediator) hypergraph)
     (= (SET.FTN$target comediator) set.ftn$function)
     (forall (?g (hypergraph ?g))
         (and (= (set.ftn$source (comediator ?g)) (case ?g))
              (= (set.ftn$target (comediator ?g)) (node ?g))
              (= (comediator ?g)
                 (set.ftn$composition
                     [(case-tuple ?g) (set.pr$comediator (reference ?g))])))))
```

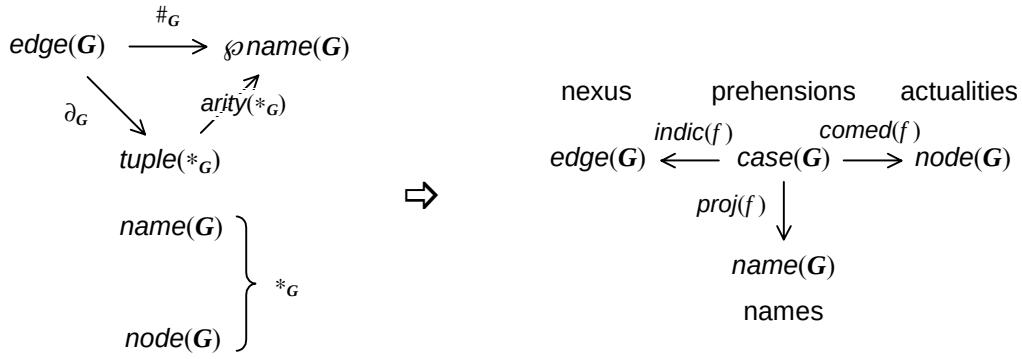


Diagram 3: From hypergraph to spangraph

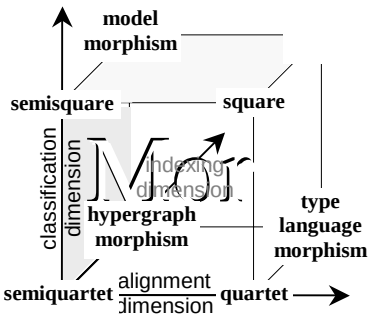
- Associated with any hypergraph $\boldsymbol{G}$ is a spangraph $sgph(\boldsymbol{G}) = \langle 1^{st}_{sgph(\boldsymbol{G})}, 2^{nd}_{sgph(\boldsymbol{G})}, 3^{rd}_{sgph(\boldsymbol{G})} \rangle$ (Diagram 3), whose vertex set is the case set of $\boldsymbol{G}$, and whose first (actualities), second (prehensions) and third (nexus) functions are

  $1^{st}_{sgph(\boldsymbol{G})} = comed(\boldsymbol{G}) : case(\boldsymbol{G}) \to node(\boldsymbol{G})$,

  $2^{nd}_{sgph(\boldsymbol{G})} = proj(\boldsymbol{G}) : case(\boldsymbol{G}) \to name(\boldsymbol{G})$, and

  $3^{st}_{sgph(\boldsymbol{G})} = indic(\boldsymbol{G}) : case(\boldsymbol{G}) \to edge(\boldsymbol{G})$.

```
(18) (SET.FTN$function spangraph)
     (= (SET.FTN$source spangraph) hypergraph)
     (= (SET.FTN$target spangraph) sgph$graph)
     (forall (?g (hypergraph ?g))
         (and (= (sgph$vertex (spangraph ?g)) (case ?g))
              (= (sgph$first (spangraph ?g)) (comediator ?g))
              (= (sgph$set1 (spangraph ?g)) (node ?g))
              (= (sgph$second (spangraph ?g)) (projection ?g))
              (= (sgph$set2 (spangraph ?g)) (name ?g))
              (= (sgph$third (spangraph ?g)) (indication ?g))
              (= (sgph$set3 (spangraph ?g)) (edge ?g))))
```



Hypergraph Morphisms

`hgph.mor`

Hypergraphs are connected by and comparable with hypergraph morphisms. This section discusses hypergraph morphisms. First, we give a concise mathematical definition, and then we discuss and formalize the various parts of this definition.

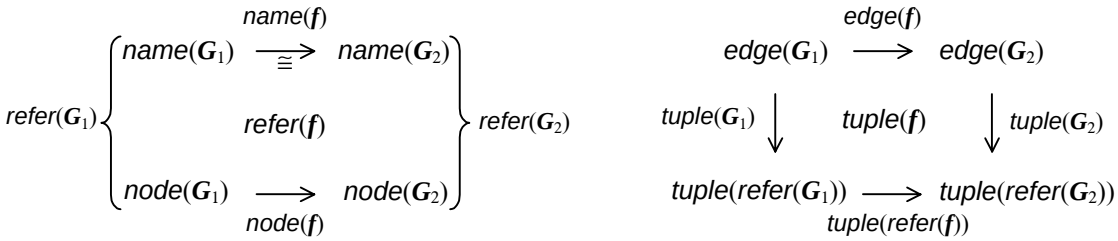


Figure 5: Hypergraph Morphism

- A *hypergraph morphism* $\boldsymbol{f} = \langle \mathit{refer}(\boldsymbol{f}), \mathit{tuple}(\boldsymbol{f}) \rangle : \boldsymbol{G}_1 \to \boldsymbol{G}_2$ from hypergraph $\boldsymbol{G}_1$ to hypergraph $\boldsymbol{G}_2$ (Figure 5) is a two dimensional construction consisting of
  - a *reference* semiquartet $*_f = \mathit{refer}(\boldsymbol{f})$ and
  - a *tuple* quartet $\mathit{tuple}(\boldsymbol{f})$,

  where the tuple function of the reference semiquartet is the vertical target of the tuple quartet

  $\mathit{tuple}(\mathit{refer}(\boldsymbol{f})) = \mathit{vert\text{-}tgt}(\mathit{tuple}(\boldsymbol{f}))$.

```
(1) (SET$class hypergraph-morphism)

(2) (SET.FTN$function source)
    (= (SET.FTN$source source) hypergraph-morphism)
    (= (SET.FTN$target source) hgph$hypergraph)

(3) (SET.FTN$function target)
    (= (SET.FTN$source target) hypergraph-morphism)
    (= (SET.FTN$target target) hgph$hypergraph)

(4) (SET.FTN$function reference)
    (= (SET.FTN$source reference) hypergraph-morphism)
    (= (SET.FTN$target reference) set.sqtt$semiquartet)
    (= (SET.FTN$composition [reference set.sqtt$source])
       (SET.FTN$composition [source hgph$reference]))
    (= (SET.FTN$composition [reference set.sqtt$target])
       (SET.FTN$composition [target hgph$reference]))

(5) (SET.FTN$function tuple)
    (= (SET.FTN$source tuple) hypergraph-morphism)
    (= (SET.FTN$target tuple) set.qtt$quartet)
    (= (SET.FTN$composition [tuple set.qtt$horizontal-source])
       (SET.FTN$composition [source hgph$tuple]))
    (= (SET.FTN$composition [tuple set.qtt$horizontal-target])
       (SET.FTN$composition [target hgph$tuple]))

(6) (= (SET.FTN$composition [tuple set.qtt$vertical-target])
       (SET.FTN$composition [reference set.sqtt$tuple]))
```

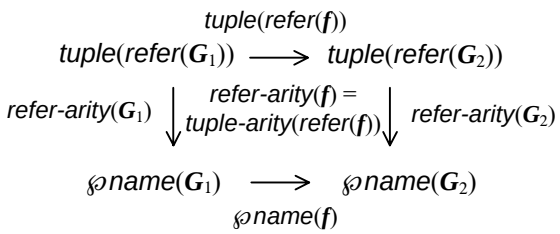


Figure 6: Reference-arity quartet

- For convenience of theoretical presentation, we introduce additional hypergraph terminology for the composition between the reference function to semiquartets and the tuple arity function from semi-quartets to quartets. Associated with a hypergraph morphism $\boldsymbol{f} : \boldsymbol{G}_1 \to \boldsymbol{G}_2$ is a *reference-arity* quartet *refer-arity*(***f***).

```
(7) (SET.FTN$function reference-arity)
    (= (SET.FTN$source reference-arity) hypergraph-morphism)
    (= (SET.FTN$target reference-arity) set.qtt$quartet)
    (= reference-arity (SET.FTN$composition [reference set.sqtt$tuple-arity]))
```

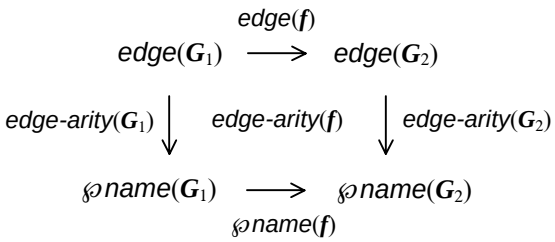


Figure 7: Edge-arity quartet

- The vertical composition of the tuple quartet with the reference-arity quartet defines the *edge arity* quartet $\#_f$ = *edge-arity*(***f***) = *tuple*(***f***) • *refer-arity*(***f***).

```
(8) (SET.FTN$function edge-arity)
    (= (SET.FTN$source edge-arity) hypergraph-morphism)
    (= (SET.FTN$target edge-arity) set.qtt$quartet)
    (forall (?f (hypergraph-morphism ?f))
         (= (edge-arity ?f)
            (set.qtt$vertical-composition [(tuple ?f) (reference-arity ?f)])))
```

- For convenience of reference, we introduce additional hypergraph terminology for the vertical source (function_1) and target components (function_2) of these (semi)quartets. The vertical source of the tuple quartet is called the *edge* function of ***f*** and denoted *edge*(***f***). The second function of the reference semiquartet is called the *node* function of ***f*** and denoted *node*(***f***). The first function (bijection) of the reference semiquartet is called the *name* bijection of ***f*** and denoted *name*(***f***). In summary, a hypergraph morphism has the following component functions:
  - the *node* function $\mathit{node}(f) = \mathit{fn2}(\mathit{refer}(f)) : \mathit{node}(\boldsymbol{G}_1) \to \mathit{node}(\boldsymbol{G}_2)$,
  - the *edge* function $\mathit{edge}(f) = \mathit{vert\text{-}src}(\mathit{tuple}(f)) : \mathit{edge}(\boldsymbol{G}_1) \to \mathit{edge}(\boldsymbol{G}_2)$, and
  - the *name* bijection $\mathit{name}(f) = \mathit{fn1}(\mathit{refer}(f)) : \mathit{name}(\boldsymbol{G}_1) \to \mathit{name}(\boldsymbol{G}_2)$.

  This results in the following presentations of the reference semiquartet and the tuple and arity quartets:
  - the *tuple* quartet $\mathit{tuple}(\boldsymbol{f}) = \langle \mathit{edge}(\alpha), \mathit{tuple}(\mathit{refer}(\boldsymbol{G}))\rangle : \mathit{tuple}(\mathit{src}(\boldsymbol{f})) \to \mathit{tuple}(\mathit{tgt}(\boldsymbol{f}))$,
  - the *edge-arity* quartet

    $\mathit{edge\text{-}arity}(\boldsymbol{f}) = \langle \mathit{edge}(\boldsymbol{f}),\ \wp\mathit{name}(\boldsymbol{f})\rangle : \mathit{edge\text{-}arity}(\mathit{src}(\alpha)) \to \mathit{edge\text{-}arity}(\mathit{tgt}(\alpha))$, and
  - the *reference* semiquartet $\mathit{refer}(\boldsymbol{f}) = \langle \mathit{name}(\boldsymbol{f}), \mathit{node}(\boldsymbol{f})\rangle : \mathit{refer}(\mathit{src}(\alpha)) \to \mathit{refer}(\mathit{tgt}(\alpha))$.

Therefore, a hypergraph morphism can be displayed as in Figure 5.

The edge-arity quartet asserts the preservation of hyperedge arity: $\#_{G1} \cdot \wp\mathit{name}(f) = \mathit{edge}(f) \cdot \#_{G2}$. This means that the direct-image of the $\boldsymbol{G}_1$-arity of any hyperedge $\varepsilon \in \mathit{edge}(\boldsymbol{G}_1)$ is the $\boldsymbol{G}_2$-arity of the image of the hyperedge: $\wp\mathit{name}(f)(\#_{G1}(\varepsilon)) = \#_{G2}(\mathit{edge}(f)(\varepsilon))$; or symbolically,

if $(\varepsilon, x_1, x_2, \ldots x_n)$ then $(\mathit{edge}(f)(\varepsilon), \mathit{name}(f)(x_1), \mathit{name}(f)(x_2), \ldots \mathit{name}(f)(x_n))$.

The tuple quartet asserts the preservation of hyperedge tuple: $\partial_{G1} \cdot \mathit{tuple}(\mathit{refer}(f)) = \mathit{edge}(f) \cdot \partial_{G2}$. This means that the image of the $\boldsymbol{G}_1$-tuple of any hyperedge $\varepsilon \in \mathit{edge}(\boldsymbol{G}_1)$ is the $\boldsymbol{G}_2$-tuple of the image of the hyperedge: $\mathit{tuple}(\mathit{refer}(f))(\partial_{G1}(\varepsilon)) = \partial_{G2}(\mathit{edge}(f)(\varepsilon))$; or symbolically,

if $(\varepsilon, x_1 : \nu_1, x_2 : \nu_2, \ldots x_n : \nu_n)$

then $(\mathit{edge}(f)(\varepsilon), \mathit{name}(f)(x_1) : \mathit{node}(f)(\nu_1), \ldots \mathit{name}(f)(x_n) : \mathit{node}(f)(\nu_n))$.

Since the tuple function $\mathit{tuple}(\mathit{refer}(f))$ preserves arity up to isomorphism, the edge function $\mathit{edge}(f)$ also does this.

```
(9) (SET.FTN$function node)
    (= (SET.FTN$source node) hypergraph-morphism)
    (= (SET.FTN$target node) set.ftn$function)
    (= node (SET.FTN$composition [reference set.sqtt$function2]))

(10) (SET.FTN$function edge)
     (= (SET.FTN$source edge) hypergraph-morphism)
     (= (SET.FTN$target edge) set.ftn$function)
     (= edge (SET.FTN$composition [tuple set.qtt$vertical-source]))

(11) (SET.FTN$function name)
     (= (SET.FTN$source name) hypergraph-morphism)
     (= (SET.FTN$target name) set.ftn$function)
     (= name (SET.FTN$composition [reference set.sqtt$function1]))
```

- Two hypergraph morphisms are *composable* when the target of the first is equal to the source of the second. The *composition* of two composable hypergraph morphisms $\boldsymbol{f}_1 : \boldsymbol{G} \to \boldsymbol{G}'$ and $\boldsymbol{f}_2 : \boldsymbol{G}' \to \boldsymbol{G}''$ is defined in terms of the horizontal composition of their tuple quartet and reference semiquartet.

```
(12) (SET.LIM.PBK$opspan composable-opspan)
     (= (class1 composable-opspan) hypergraph-morphism)
     (= (class2 composable-opspan) hypergraph-morphism)
     (= (opvertex composable-opspan) hgph$hypergraph)
     (= (first composable-opspan) target)
     (= (second composable-opspan) source)

(13) (REL$relation composable)
     (= (REL$class1 composable) hypergraph-morphism)
     (= (REL$class2 composable) hypergraph-morphism)
     (= (REL$extent composable) (SET.LIM.PBK$pullback composable-opspan))

(14) (SET.FTN$function composition)
     (= (SET.FTN$source composition) (SET.LIM.PBK$pullback composable-opspan))
     (= (SET.FTN$target composition) hypergraph-morphism)
     (forall (?f1 (hypergraph-morphism ?f1) ?f2 (hypergraph-morphism ?f2)
             (composable ?f1 ?f2))
         (and (= (source (composition [?f1 ?f2])) (source ?f1))
              (= (target (composition [?f1 ?f2])) (target ?f2))
              (= (reference (composition [?f1 ?f2]))
                 (set.sqtt$composition [(reference ?f1) (reference ?f2)]))
              (= (tuple (composition [?f1 ?f2]))
                 (set.qtt$horizontal-composition [(tuple ?f1) (tuple ?f2)])))))
```

- Composition satisfies the usual *associative law*.

```
(forall (?f1 (hypergraph-morphism ?f1)
         ?f2 (hypergraph-morphism ?f2)
         ?f3 (hypergraph-morphism ?f3)
         (composable ?f1 ?f2) (composable ?f2 ?f3))
    (= (composition [?f1 (composition [?f2 ?f3])])
       (composition [(composition [?f1 ?f2]) ?f3])))
```

- For any hypergraph ***G***, there is an *identity* hypergraph morphism.

```
(15) (SET.FTN$function identity)
     (= (SET.FTN$source identity) hgph$hypergraph)
     (= (SET.FTN$target identity) hypergraph-morphism)
     (forall (?g (hgph$hypergraph ?g))
         (and (= (source (identity ?g)) ?g)
              (= (target (identity ?g)) ?g)
              (= (reference (identity ?g))
                 (set.sqtt$identity (hgph$reference ?g)))
              (= (tuple (identity ?g))
                 (set.qtt$horizontal-identity (hgph$tuple ?g)))))
```

- The identity satisfies the usual *identity laws* with respect to composition.

```
(forall (?f (hypergraph-morphism ?f))
    (and (= (composition [(identity (source ?f)) ?f]) ?f)
         (= (composition [?f (identity (target ?f))]) ?f)))
```

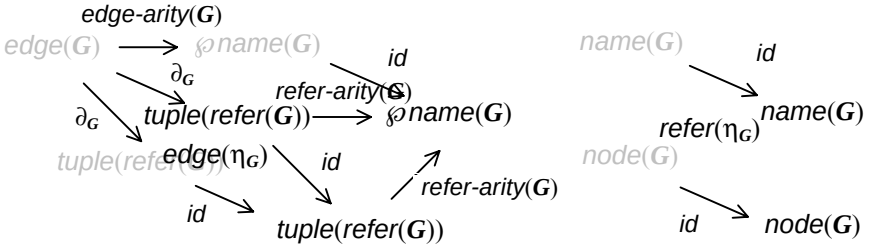


Figure 8: Eta Hypergraph Morphism

- For every hypergraph ***G***, there is an special hypergraph morphism (Figure 8)

  $\eta_{G} : G \to hgph(refer(G))$

  from ***G*** to the hypergraph of its reference pair, where

  $refer(\eta_{G}) = id_{refer(G)}$, $vert\text{-}src(edge(\eta_{G})) = tuple(G)$,

  and $vert\text{-}tgt(edge(\eta_{G})) = id_{tuple(refer(G))}$.

```
(16) (SET.FTN$function eta)
    (= (SET.FTN$source eta) hgph$hypergraph)
    (= (SET.FTN$target eta) hypergraph-morphism)
    (forall (?g (hgph$hypergraph ?g))
        (and (= (source (eta ?g)) ?g)
             (= (target (eta ?g)) (set.sqtt$hypergraph (hgph$reference ?g)))
             (= (reference (eta ?g))
                (set.sqtt$identity (hgph$reference ?g)))
             (= (set.qtt$vertical-source (tuple (eta ?g)))
                (hgph$tuple ?g))
             (= (set.qtt$vertical-target (tuple (eta ?g)))
                (set.ftn$semi-identity (hgph$tuple ?g)))))
```

Case and Spangraph Morphisms

- Any hypergraph morphism $\boldsymbol{f} = \langle refer(\boldsymbol{f}), tuple(\boldsymbol{f}) \rangle : \boldsymbol{G}_1 \to \boldsymbol{G}_2$ defines a coproduct *arity morphism*, whose set quartet is edge arity

  $edge\text{-}arity(\boldsymbol{f}) = \langle edge(\boldsymbol{f}), \wp name(\boldsymbol{f}) \rangle : edge\text{-}arity(\boldsymbol{G}_1) \to edge\text{-}arity(\boldsymbol{G}_2)$.

```
(17) (SET.FTN$function arity-morphism)
     (= (SET.FTN$source arity-morphism) hypergraph-morphism)
     (= (SET.FTN$target arity-morphism) set.col.art.mor$arity-morphism)
     (forall (?f (hypergraph-morphism ?f))
          (and (= (set.col.art.mor$source (arity-morphism ?f)) (hgph$arity (source ?f)))
               (= (set.col.art.mor$target (arity-morphism ?f)) (hgph$arity (target ?f)))
               (= (set.col.art.mor$index (arity-morphism ?f)) (edge ?f))
               (= (set.col.art.mor$base (arity-morphism ?f)) (name ?f))))
```

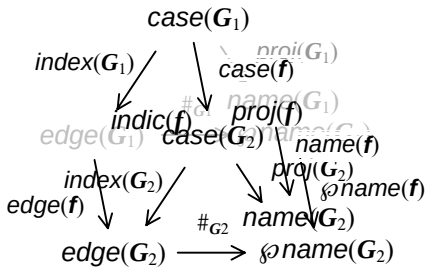


Figure 9: Case function = Coproduct of the Arity of a Hypergraph Morphism

- Any hypergraph morphism $\boldsymbol{f} = \langle refer(\boldsymbol{f}), tuple(\boldsymbol{f}) \rangle : \boldsymbol{G}_1 \to \boldsymbol{G}_2$ defines a *case* function $case(f) = \sum arity(\boldsymbol{f}) : case(G_1) = \sum arity(\boldsymbol{G}_1) \to \sum arity(\boldsymbol{G}_2) = case(\boldsymbol{G}_2)$. The pointwise definition is:

  $case(\boldsymbol{f})((\varepsilon, x)) = (edge(f)(\varepsilon), name(f)(x))$

  for any hyperedge $\varepsilon \in edge(\boldsymbol{G}_1)$ and any name $x \in arity(\boldsymbol{G}_1)(\varepsilon)$. This is well defined since $\boldsymbol{f}$ preserves hyperedge arity. The abstract definition (Figure 9) is in terms of the coproduct of the arity morphism.

```
(18) (SET.FTN$function case)
     (= (SET.FTN$source case) hypergraph-morphism)
     (= (SET.FTN$target case) set.ftn$function)
     (forall (?f (hypergraph-morphism ?f))
         (and (= (set.ftn$source (case ?f)) (hgph$case (source ?f)))
              (= (set.ftn$target (case ?f)) (hgph$case (target ?f)))
              (= (case ?f)
                 (set.col.art.mor$coproduct (arity-morphism ?f)))))
```

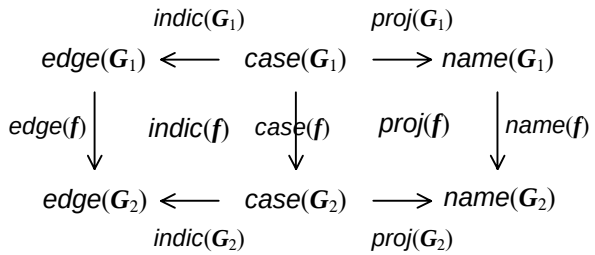
- 

Figure 10: Indication and Projection Quartets

- The case function is the vertical source for two quartets (Figure 10):
  an *indication* quartet $indic(\boldsymbol{f})$ and a *projection* quartet $proj(\boldsymbol{f})$.
  - The commutativity $case(f) \cdot indic(\boldsymbol{G}_2) = indic(\boldsymbol{G}_1) \cdot edge(\boldsymbol{f})$, a property of the coproduct of arities (preservation of index), is obvious from the pointwise definition of the case function.

– The commutativity $case(f) \cdot proj(\boldsymbol{G}_2) = proj(\boldsymbol{G}_1) \cdot name(\boldsymbol{f})$, a property of the coproduct of arities (preservation of projection), is obvious from the pointwise definition of the case function.

Even though the name function $name(f) : name(\boldsymbol{G}_1) \to name(\boldsymbol{G}_2)$ and its power $\wp name(f)$ are bijections, the case function $case(f) : case(\boldsymbol{G}_1) \to case(\boldsymbol{G}_2)$ is not necessarily a bijection since the edge function $edge(f) : edge(\boldsymbol{G}_1) \to edge(\boldsymbol{G}_2)$ need not be bijective. By the preservation of hyperedge arity, the index quartet is a fibration: for any hyperedge $\varepsilon_1 \in edge(\boldsymbol{G}_1)$ and any name $x_2 \in arity(\boldsymbol{G}_2)(edge(f)(\varepsilon_1))$ there is a name $x_1 \in arity(\boldsymbol{G}_1)(\varepsilon_1)$ such that $name(f)(x_1) = x_2$.

```
(19) (SET.FTN$function indication)
     (= (SET.FTN$source indication) hypergraph-morphism)
     (= (SET.FTN$target indication) set.qtt$fibration)
     (forall (?f (hypergraph-morphism ?f))
         (and (= (set.qtt$horizontal-source (indication ?f)) (hgph$indication (source ?f)))
              (= (set.qtt$horizontal-target (indication ?f)) (hgph$indication (target ?f)))
              (= (set.qtt$vertical-source (indication ?f)) (case ?f))
              (= (set.qtt$vertical-target (indication ?f)) (edge ?f))))

(20) (SET.FTN$function projection)
     (= (SET.FTN$source projection) hypergraph-morphism)
     (= (SET.FTN$target projection) set.qtt$quartet)
     (forall (?f (hypergraph-morphism ?f))
         (and (= (set.qtt$horizontal-source (projection ?f)) (hgph$projection (source ?f)))
              (= (set.qtt$horizontal-target (projection ?f)) (hgph$projection (target ?f)))
              (= (set.qtt$vertical-source (projection ?f)) (case ?f))
              (= (set.qtt$vertical-target (projection ?f)) (name ?f))))
```

$$\begin{array}{ccc} & \tilde{*}_{\boldsymbol{G}1} = comed(\boldsymbol{G}_1) & \\ case(\boldsymbol{G}_1) & \longrightarrow & node(\boldsymbol{G}_1) \\ \Big\downarrow{\scriptstyle case(\boldsymbol{f})} & comed(\boldsymbol{f}) & \Big\downarrow{\scriptstyle node(\boldsymbol{f})} \\ case(\boldsymbol{G}_2) & \longrightarrow & node(\boldsymbol{G}_2) \\ & \tilde{*}_{\boldsymbol{G}2} = comed(\boldsymbol{G}_2) & \end{array}$$

Figure 11: Comediator Quartet

○ Any hypergraph morphism $\boldsymbol{f} = \langle refer(\boldsymbol{f}), tuple(\boldsymbol{f}) \rangle : \boldsymbol{G}_1 \to \boldsymbol{G}_2$ defines a *comediator* quartet $\tilde{*}_f = comed(\boldsymbol{f})$ (Figure 11). The commutativity $case(f) \cdot comed(\boldsymbol{G}_2) = comed(\boldsymbol{G}_1) \cdot node(\boldsymbol{f})$ holds by a property of the coproduct of arities (preservation of cotupling). Commutativity states that

$$\partial_{\boldsymbol{G}2}(edge(f)(\varepsilon))(name(f)(x)) = node(f)(\partial_{\boldsymbol{G}1}(\varepsilon)(x))$$

for any hyperedge $\varepsilon \in edge(\boldsymbol{G}_1)$ and any name $x \in name(\boldsymbol{G}_1)$, which is true by preservation of hyperedge tuple.

```
(21) (SET.FTN$function comediator)
     (= (SET.FTN$source comediator) hypergraph-morphism)
     (= (SET.FTN$target comediator) set.qtt$quartet)
     (forall (?f (hypergraph-morphism ?f))
         (and (= (set.qtt$horizontal-source (comediator ?f)) (hgph$comediator (source ?f)))
              (= (set.qtt$horizontal-target (comediator ?f)) (hgph$comediator (target ?f)))
              (= (set.qtt$vertical-source (comediator ?f)) (case ?f))
              (= (set.qtt$vertical-target (comediator ?f)) (node ?f))))
```

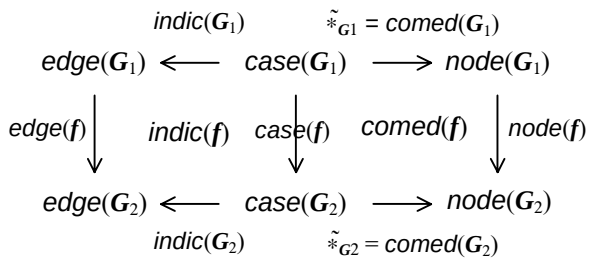


Figure 12: Spangraph Morphism of a Hypergraph Morphism

- Associated with any hypergraph morphism $\boldsymbol{f} = \langle \mathit{refer}(\boldsymbol{f}), \mathit{tuple}(\boldsymbol{f}) \rangle : \boldsymbol{G}_1 \to \boldsymbol{G}_2$ is a spangraph morphism
  $\mathit{sgph\text{-}mor}(\boldsymbol{f}) = \langle 1^{st}_{\mathit{sgph}(\boldsymbol{f})}, 2^{nd}_{\mathit{sgph}(\boldsymbol{f})}, 3^{rd}_{\mathit{sgph}(\boldsymbol{f})} \rangle : \mathit{sgph}(\boldsymbol{G}_1) \to \mathit{sgph}(\boldsymbol{G}_2)$ (Figure 12),
  whose vertex function is the case functions, and whose three quartets (the 3rd is a fibration) are
  $1^{st}_{\mathit{sgph}(\boldsymbol{f})}(f) = \mathit{comed}(\boldsymbol{f})$,
  $2^{nd}_{\mathit{sgph}(\boldsymbol{f})}(f) = \mathit{proj}(\boldsymbol{f})$, and
  $3^{rd}_{\mathit{sgph}(\boldsymbol{f})}(f) = \mathit{indic}(\boldsymbol{f})$.

```
(22) (SET.FTN$function spangraph-morphism)
     (= (SET.FTN$source spangraph-morphism) hypergraph-morphism)
     (= (SET.FTN$target spangraph-morphism) sgph.mor$spangraph-morphism)
     (forall (?f (hypergraph-morphism ?f))
         (and (= (sgph.mor$source (spangraph-morphism ?f)) (hgph$spangraph (source ?f)))
              (= (sgph.mor$target (spangraph-morphism ?f)) (hgph$spangraph (target ?f)))
              (= (sgph.mor$first (spangraph-morphism ?f)) (comediator ?f))
              (= (sgph.mor$second (spangraph-morphism ?f)) (projection ?f))
              (= (sgph.mor$third (spangraph-morphism ?f)) (indication ?f))))
```