

Hypergraph Basics

Chelsea Edmonds and Lawrence C. Paulson

April 18, 2024

Abstract

This entry is a simple extension of our previous entry for Combinatorial design theory [1], which presents new and existing concepts using hypergraph language. Both designs and hypergraphs are types of incident set systems, hence have the same underlying foundation. However, they are often used in different contexts, and some definitions are as such unique. This library uses locales to rewrite equivalent definitions and build a basic hypergraph hierarchy with direct links to equivalent design theory concepts to avoid repetition, further demonstrating the power of the “locale-centric” approach. The library includes all standard definitions (order, degree etc.), as well as some extensions on hypergraph decompositions and spanning subhypergraphs.

Contents

1 Basic Hypergraphs	1
1.1 Sub hypergraphs	4
2 Hypergraph Variations	5
2.1 Non-trivial hypergraphs	5
2.2 Regular and Uniform Hypergraphs	7
2.3 Factorisations	8
2.4 Sample Graph Theory Connections	8

1 Basic Hypergraphs

Converting Design theory to hypergraph notation. Hypergraphs have technically already been formalised

```
theory Hypergraph
imports
  Design-Theory.Block-Designs
  Design-Theory.Sub-Designs
  Fishers-Inequality.Design-Extras
begin
```

lemma *is-singleton-image*:
is-singleton $C \implies \text{is-singleton } (f \text{ ' } C)$
 ⟨*proof*⟩

lemma *bij-betw-singleton-image*:
assumes *bij-betw* $f A B$
assumes $C \subseteq A$
shows *is-singleton* $C \iff \text{is-singleton } (f \text{ ' } C)$
 ⟨*proof*⟩

lemma *image-singleton*:
assumes $A \neq \{\}$
assumes $\bigwedge x. x \in A \implies f x = c$
shows $f \text{ ' } A = \{c\}$
 ⟨*proof*⟩

type-synonym *colour* = *nat*

type-synonym *'a hyp-edge* = *'a set*

type-synonym *'a hyp-graph* = (*'a set*) \times (*'a hyp-edge multiset*)

abbreviation *hyp-edges* :: *'a hyp-graph* \Rightarrow *'a hyp-edge multiset* **where**
hyp-edges $H \equiv \text{snd } H$

abbreviation *hyp-verts* :: *'a hyp-graph* \Rightarrow *'a set* **where**
hyp-verts $H \equiv \text{fst } H$

locale *hypersystem* = *incidence-system* *vertices* :: *'a set* *edges* :: *'a hyp-edge multiset*

for *vertices* (\mathcal{V}) **and** *edges* (E)

begin

Basic definitions using hypergraph language

abbreviation *horder* :: *nat* **where**
horder $\equiv \text{card } (\mathcal{V})$

definition *hdegree* :: *'a* \Rightarrow *nat* **where**
hdegree $v \equiv \text{size } \{\#e \in \# E . v \in e \#\}$

lemma *hdegree-rep-num*: *hdegree* $v = \text{point-replication-number } E v$
 ⟨*proof*⟩

definition *hdegree-set* :: *'a set* \Rightarrow *nat* **where**
hdegree-set $vs \equiv \text{size } \{\#e \in \# E . vs \subseteq e \#\}$

lemma *hdegree-set-points-index*: *hdegree-set* $vs = \text{points-index } E vs$

<proof>

definition *hvert-adjacent* :: 'a \Rightarrow 'a \Rightarrow bool **where**

hvert-adjacent v1 v2 $\equiv \exists e . e \in \# E \wedge v1 \in e \wedge v2 \in e \wedge v1 \in \mathcal{V} \wedge v2 \in \mathcal{V}$

definition *hedge-adjacent* :: 'a *hyp-edge* \Rightarrow 'a *hyp-edge* \Rightarrow bool **where**

hedge-adjacent e1 e2 $\equiv e1 \cap e2 \neq \{\}$ $\wedge e1 \in \# E \wedge e2 \in \# E$

lemma *edge-adjacent-alt-def*: e1 $\in \# E \implies e2 \in \# E \implies \exists x . x \in \mathcal{V} \wedge x \in e1$
 $\wedge x \in e2 \implies$

hedge-adjacent e1 e2

<proof>

definition *hneighborhood* :: 'a \Rightarrow 'a set **where**

hneighborhood x $\equiv \{v \in \mathcal{V} . \text{hvert-adjacent } x v\}$

definition *hmax-degree* :: nat **where**

hmax-degree $\equiv \text{Max } \{hdegree v \mid v . v \in \mathcal{V}\}$

definition *hrank* :: nat **where**

hrank $\equiv \text{Max } \{\text{card } e \mid e . e \in \# E\}$

definition *hcorank* :: nat **where**

hcorank = *Min* $\{\text{card } e \mid e . e \in \# E\}$

definition *hedge-neighbourhood* :: 'a \Rightarrow 'a *hyp-edge* multiset **where**

hedge-neighbourhood x $\equiv \{\# e \in \# E . x \in e \#\}$

lemma *degree-alt-neighbourhood*: *hdegree* x = *size* (*hedge-neighbourhood* x)

<proof>

definition *hinduced-edges*:: 'a set \Rightarrow 'a *hyp-edge* multiset **where**

hinduced-edges V' = $\{\#e \in \# E . e \subseteq V'\#\}$

end

Sublocale for rewriting definition purposes rather than inheritance

sublocale *hypersystem* \subseteq *incidence-system* $\mathcal{V} E$

rewrites *point-replication-number* E v = *hdegree* v **and** *points-index* E vs =
hdegree-set vs

<proof>

Reverse sublocale to establish equality

sublocale *incidence-system* \subseteq *hypersystem* $\mathcal{V} \mathcal{B}$

rewrites *hdegree* v = *point-replication-number* \mathcal{B} v **and** *hdegree-set* vs = *points-index*
 \mathcal{B} vs

<proof>

Missing design identified in the design theory hierarchy

locale *inf-design* = *incidence-system* +

assumes *blocks-nempty*: $bl \in \# \mathcal{B} \implies bl \neq \{\}$
sublocale *design* \subseteq *inf-design*
 ⟨*proof*⟩
locale *fin-hypersystem* = *hypersystem* + *finite-incidence-system* \mathcal{V} *E*
sublocale *finite-incidence-system* \subseteq *fin-hypersystem* \mathcal{V} \mathcal{B}
 ⟨*proof*⟩
locale *hypergraph* = *hypersystem* + *inf-design* \mathcal{V} *E*
sublocale *inf-design* \subseteq *hypergraph* \mathcal{V} \mathcal{B}
 ⟨*proof*⟩
locale *fin-hypergraph* = *hypergraph* + *fin-hypersystem*
sublocale *design* \subseteq *fin-hypergraph* \mathcal{V} \mathcal{B}
 ⟨*proof*⟩
sublocale *fin-hypergraph* \subseteq *design* \mathcal{V} *E*
 ⟨*proof*⟩

1.1 Sub hypergraphs

Sub hypergraphs and related concepts (spanning hypergraphs etc)

locale *sub-hypergraph* = *sub*: *hypergraph* $\mathcal{V}H$ *EH* + *orig*: *hypergraph* $\mathcal{V} :: 'a$ set *E*
 +
sub-set-system $\mathcal{V}H$ *EH* \mathcal{V} *E* **for** $\mathcal{V}H$ *EH* \mathcal{V} *E*

locale *spanning-hypergraph* = *sub-hypergraph* +
assumes $\mathcal{V} = \mathcal{V}H$

lemma *spanning-hypergraphI*: *sub-hypergraph* $\mathcal{V}H$ *EH* \mathcal{V} *E* $\implies \mathcal{V} = \mathcal{V}H \implies$
spanning-hypergraph $\mathcal{V}H$ *EH* \mathcal{V} *E*
 ⟨*proof*⟩

context *hypergraph*
begin

definition *is-subhypergraph* :: '*a* *hyp-graph* \Rightarrow *bool* **where**
is-subhypergraph *H* \equiv *sub-hypergraph* (*hyp-verts* *H*) (*hyp-edges* *H*) \mathcal{V} *E*

lemma *is-subhypergraphI*:
assumes (*hyp-verts* *H* \subseteq \mathcal{V})
assumes (*hyp-edges* *H* $\subseteq \#$ *E*)
assumes *hypergraph* (*hyp-verts* *H*) (*hyp-edges* *H*)
shows *is-subhypergraph* *H*
 ⟨*proof*⟩

definition *hypergraph-decomposition* :: 'a hyp-graph multiset \Rightarrow bool **where**
hypergraph-decomposition $S \equiv (\forall h \in \# S . \text{is-subhypergraph } h) \wedge$
partition-on-mset $E \{\# \text{hyp-edges } h . h \in \# S \# \}$

definition *is-spanning-subhypergraph* :: 'a hyp-graph \Rightarrow bool **where**
is-spanning-subhypergraph $H \equiv \text{spanning-hypergraph } (\text{hyp-verts } H) (\text{hyp-edges } H)$
 $\mathcal{V} E$

lemma *is-spanning-subhypergraphI*: *is-subhypergraph* $H \Longrightarrow (\text{hyp-verts } H) = \mathcal{V}$
 \Longrightarrow
is-spanning-subhypergraph H
 <proof>

lemma *spanning-subhypergraphI*: $(\text{hyp-verts } H) = \mathcal{V} \Longrightarrow (\text{hyp-edges } H) \subseteq \# E$
 \Longrightarrow
hypergraph $(\text{hyp-verts } H) (\text{hyp-edges } H) \Longrightarrow \text{is-spanning-subhypergraph } H$
 <proof>

end
end

2 Hypergraph Variations

This section presents many different types of hypergraphs, introducing conditions such as non-triviality, regularity, and uniform. Additionally, it briefly formalises decompositions

theory *Hypergraph-Variations*
imports
Hypergraph
Undirected-Graph-Theory.Bipartite-Graphs
begin

2.1 Non-trivial hypergraphs

Non empty (ne) implies that the vertex (and edge) set is not empty. Non trivial typically requires at least two edges

locale *hyper-system-vne* = *hypersystem* +
assumes *V-nonempty*: $\mathcal{V} \neq \{\}$

locale *hyper-system-ne* = *hyper-system-vne* +
assumes *E-nonempty*: $E \neq \{\# \}$

locale *hypergraph-ne* = *hypergraph* +
assumes *E-nonempty*: $E \neq \{\# \}$
begin

lemma *V-nempty*: $\mathcal{V} \neq \{\}$
 ⟨*proof*⟩

lemma *sizeE-not-zero*: $\text{size } E \neq 0$
 ⟨*proof*⟩

end

sublocale *hypergraph-ne* \subseteq *hyper-system-ne*
 ⟨*proof*⟩

locale *hyper-system-ns* = *hypersystem* +
assumes *V-not-single*: $\neg \text{is-singleton } \mathcal{V}$

locale *hypersystem-nt* = *hyper-system-ne* + *hyper-system-ns*

locale *hypergraph-nt* = *hypergraph-ne* + *hyper-system-ns*

sublocale *hypergraph-nt* \subseteq *hypersystem-nt*
 ⟨*proof*⟩

locale *fin-hypersystem-vne* = *fin-hypersystem* + *hyper-system-vne*
begin

lemma *order-gt-zero*: $\text{horder} > 0$
 ⟨*proof*⟩

lemma *order-ge-one*: $\text{horder} \geq 1$
 ⟨*proof*⟩

end

locale *fin-hypersystem-nt* = *fin-hypersystem-vne* + *hypersystem-nt*
begin

lemma *order-gt-one*: $\text{horder} > 1$
 ⟨*proof*⟩

lemma *order-ge-two*: $\text{horder} \geq 2$
 ⟨*proof*⟩

end

locale *fin-hypergraph-ne* = *fin-hypergraph* + *hypergraph-ne*

sublocale *fin-hypergraph-ne* \subseteq *fin-hypersystem-vne*
 ⟨*proof*⟩

locale *fin-hypergraph-nt* = *fin-hypergraph* + *hypergraph-nt*

sublocale *fin-hypergraph-nt* \subseteq *fin-hypersystem-nt*
<proof>

sublocale *fin-hypergraph-ne* \subseteq *proper-design* $\mathcal{V} E$
<proof>

sublocale *proper-design* \subseteq *fin-hypergraph-ne* $\mathcal{V} \mathcal{B}$
<proof>

2.2 Regular and Uniform Hypergraphs

locale *dregular-hypergraph* = *hypergraph* +
fixes *d*
assumes *const-degree*: $\bigwedge x. x \in \mathcal{V} \implies \text{hdegree } x = d$

locale *fin-dregular-hypergraph* = *dregular-hypergraph* + *fin-hypergraph*

locale *kuniform-hypergraph* = *hypergraph* +
fixes *k :: nat*
assumes *uniform*: $\bigwedge e. e \in \# E \implies \text{card } e = k$

locale *fin-kuniform-hypergraph* = *kuniform-hypergraph* + *fin-hypergraph*

locale *almost-regular-hypergraph* = *hypergraph* +
assumes $\bigwedge x y. x \in \mathcal{V} \implies y \in \mathcal{V} \implies | \text{hdegree } x - \text{hdegree } y | \leq 1$

locale *kuniform-regular-hypergraph* = *kuniform-hypergraph* $\mathcal{V} E k$ + *dregular-hypergraph*
 $\mathcal{V} E k$
for $\mathcal{V} E k$

locale *fin-kuniform-regular-hypergraph-nt* = *kuniform-regular-hypergraph* $\mathcal{V} E k$ +
fin-hypergraph-nt $\mathcal{V} E$
for $\mathcal{V} E k$

sublocale *fin-kuniform-regular-hypergraph-nt* \subseteq *fin-kuniform-hypergraph* $\mathcal{V} E k$
<proof>

sublocale *fin-kuniform-regular-hypergraph-nt* \subseteq *fin-dregular-hypergraph* $\mathcal{V} E k$
<proof>

locale *block-balanced-design* = *block-design* + *t-wise-balance*

locale *regular-block-design* = *block-design* + *constant-rep-design*

sublocale *t-design* \subseteq *block-balanced-design*
<proof>

locale *fin-kuniform-hypergraph-nt* = *fin-kuniform-hypergraph* + *fin-hypergraph-nt*

sublocale *fin-kuniform-regular-hypgraph-nt* \subseteq *fin-kuniform-hypergraph-nt* \mathcal{V} *E k*
{*proof*}

Note that block designs are defined as non-trivial and finite as they automatically build on the proper design locale

sublocale *fin-kuniform-hypergraph-nt* \subseteq *block-design* \mathcal{V} *E k*

rewrites *point-replication-number* *E v* = *hdegree v* **and** *points-index* *E vs* = *hdegree-set vs*

{*proof*}

sublocale *fin-kuniform-regular-hypgraph-nt* \subseteq *regular-block-design* \mathcal{V} *E k k*

rewrites *point-replication-number* *E v* = *hdegree v* **and** *points-index* *E vs* = *hdegree-set vs*

{*proof*}

2.3 Factorisations

locale *d-factor* = *spanning-hypergraph* + *dregular-hypergraph* \mathcal{V} *H EH d* **for** *d*

context *hypergraph*

begin

definition *is-d-factor* :: 'a *hyp-graph* \Rightarrow *bool* **where**

is-d-factor *H* \equiv (\exists *d*. *d-factor* (*hyp-verts* *H*) (*hyp-edges* *H*) \mathcal{V} *E d*)

definition *d-factorisation* :: 'a *hyp-graph multiset* \Rightarrow *bool* **where**

d-factorisation *S* \equiv *hypergraph-decomposition* *S* \wedge (\forall *h* \in $\#$ *S*. *is-d-factor* *h*)

end

2.4 Sample Graph Theory Connections

sublocale *fin-graph-system* \subseteq *fin-hypersystem* *V mset-set E*

rewrites *hedge-adjacent* = *edge-adj*

{*proof*}

sublocale *fin-bipartite-graph* \subseteq *fin-hypersystem-vne* *V mset-set E*

{*proof*}

end

theory *Hypergraph-Basics-Root*

imports

Hypergraph

Hypergraph-Variations

begin

end

References

- [1] C. Edmonds and L. C. Paulson. Combinatorial design theory. *Archive of Formal Proofs*, August 2021. https://isa-afp.org/entries/Design_Theory.html, Formal proof development.