Task 2 & 3 – Extensions

CONTENTS

Tasks 2 & 3 – Extensions – Introduction

9 Oct 2025

What this chapter is all about

In the previous chapter we introduced **minimalist causal coding**:

- we code **links** between named factors (plain-language labels)
- we let **factor labels** do most of the work (e.g. `wealth` vs `poverty`)
- we start from a **links table** (one row per coded causal claim), rather than building a separate factor metadata structure

That approach has general applicability. This folder collects a set of **practical extensions** that make the approach more useful for answering common evaluation questions — mainly by simplifying, querying, and comparing the map.

All the extensions in this folder are implemented in the Causal Map app, but the ideas are tool-independent.

As you read:

- start with the practitioner sections (“what is this for / how do I interpret it?”)
- where a page needs more formal material, it’s placed at the bottom under **Formal notes (optional)**

Formal notes (optional)

If you want deeper theory/background (not required for using the method):

- Minimalist coding for causal mapping
- A formalisation of causal mapping

Combining links into bundles

8 Nov 2025

In most projects, the data contains many repeated causal claims with the **same cause and the same effect** (often across many sources). We call these **bundles** (or **co-terminal link bundles**).

This extension is about:

- computing **bundle-level statistics** for repeated cause->effect claims, and
- being clear that links tables and maps display bundles differently.

Why bundling is useful (for practitioners)

- **Simpler maps**: instead of 200 separate "A → B" rows, you see one "A → B" bundle with counts.
- **Clearer evidence signals**: you can read "how widely shared" (sources) vs "how often said" (citations).
- **Better reporting**: it's easy to state things like "7 sources mentioned A → B".

What you get

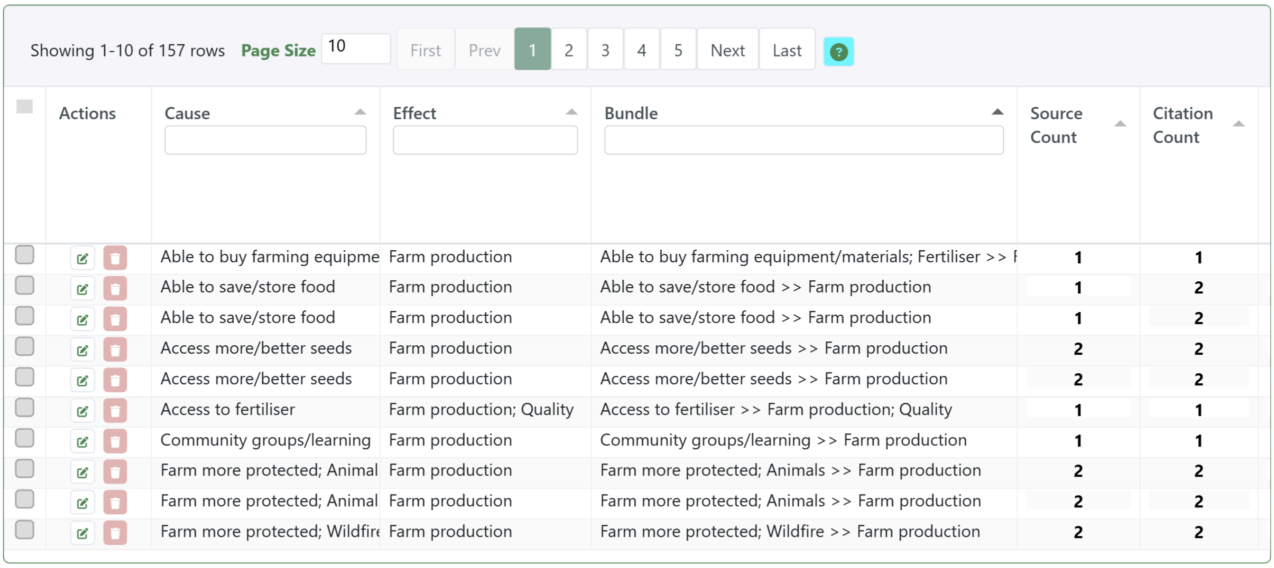
In the **links table**, each coded claim still appears as its own row (so you can still inspect individual claims/quotes separately).

What changes is that each row also gets bundle-level fields, computed from all rows with the same cause->effect pair (after transforms), for example:

- a readable bundle key like `cause >> effect`
- **source_count** (how many distinct sources made at least one claim of this form)
- **citation_count** (how many coded claims / rows are in the bundle)
- optional summaries like mean sentiment (if you use sentiment)

Rows in the same bundle share the same bundle-level values.

This means that however many filters you previously applied, including filters that might completely transform your factor labels, you still get exactly one row in the links table for every original causal claim which has not been filtered out by the filters.



Showing 1-10 of 157 rows **Page Size** 10 First Prev 1 2 3 4 5 Next Last ?

□	Actions	Cause	Effect	Bundle	Source Count	Citation Count
□		Able to buy farming equipme	Farm production	Able to buy farming equipment/materials; Fertiliser >> F	1	1
□		Able to save/store food	Farm production	Able to save/store food >> Farm production	1	2
□		Able to save/store food	Farm production	Able to save/store food >> Farm production	1	2
□		Access more/better seeds	Farm production	Access more/better seeds >> Farm production	2	2
□		Access more/better seeds	Farm production	Access more/better seeds >> Farm production	2	2
□		Access to fertiliser	Farm production; Quality	Access to fertiliser >> Farm production; Quality	1	1
□		Community groups/learning	Farm production	Community groups/learning >> Farm production	1	1
□		Farm more protected; Animal	Farm production	Farm more protected; Animals >> Farm production	2	2
□		Farm more protected; Animal	Farm production	Farm more protected; Animals >> Farm production	2	2
□		Farm more protected; Wildfir	Farm production	Farm more protected; Wildfire >> Farm production	2	2

Bookmark: #1143

How links are shown in tables vs maps

- **Links table**: one row per coded claim/citation, with repeated bundle-level stats on each row in that bundle.
- **Map view**: one visual link per bundle (one unique cause->effect pair), not one per citation. This prevents an unreadable hairball.

So if a map link label says “7 sources / 12 citations”, read it as:

- 12 coded claims were bundled into that one displayed map link, coming from
- 7 distinct sources.

This map corresponds to the links table above.

![[008 Task 2 & 3 -- Extensions/img/39412fad02d2b554610e8968d71a2d57_MD5.png|]]

Links — width: citation count; labels: source count. Filters applied: Labels: Farm production, match start. Bookmark #1144

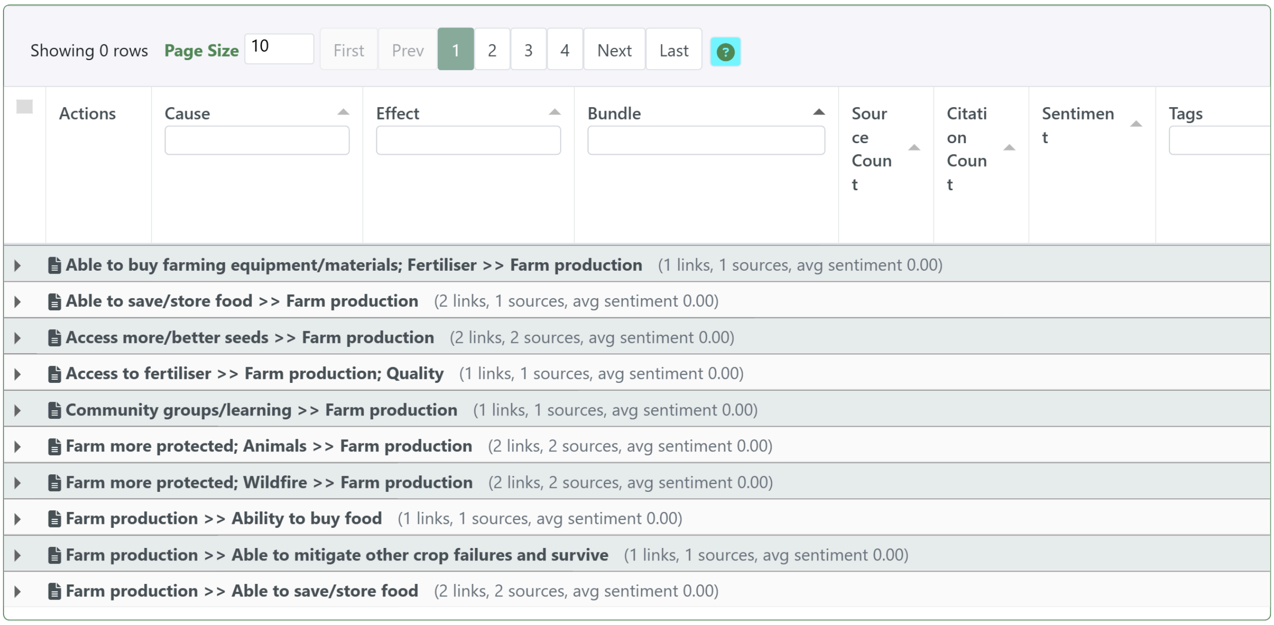
Practical cautions

- **Bundling happens after transforms**: if you zoom/collapse/combine opposites/cluster first, you are bundling the *transformed labels*, not the raw labels. That’s often what you want, but be deliberate.
- **Counts are evidence volume, not effect size**: a frequent bundle means “often claimed”, not “strong causal effect”.

You *can* view the links table grouped into bundles

In the Causal Map app, you can optionally group your links and factors and sources tables any way you want. In particular, you can group by bundle.

This view corresponds to bookmarks #1143 and #1144, above.



Showing 0 rows **Page Size** 10 First Prev 1 2 3 4 Next Last ?

□	Actions	Cause	Effect	Bundle	Source Count	Citation Count	Sentiment	Tags

▸ **Able to buy farming equipment/materials; Fertiliser >> Farm production** (1 links, 1 sources, avg sentiment 0.00)

▸ **Able to save/store food >> Farm production** (2 links, 1 sources, avg sentiment 0.00)

▸ **Access more/better seeds >> Farm production** (2 links, 2 sources, avg sentiment 0.00)

▸ **Access to fertiliser >> Farm production; Quality** (1 links, 1 sources, avg sentiment 0.00)

▸ **Community groups/learning >> Farm production** (1 links, 1 sources, avg sentiment 0.00)

▸ **Farm more protected; Animals >> Farm production** (2 links, 2 sources, avg sentiment 0.00)

▸ **Farm more protected; Wildfire >> Farm production** (2 links, 2 sources, avg sentiment 0.00)

▸ **Farm production >> Ability to buy food** (1 links, 1 sources, avg sentiment 0.00)

▸ **Farm production >> Able to mitigate other crop failures and survive** (1 links, 1 sources, avg sentiment 0.00)

▸ **Farm production >> Able to save/store food** (2 links, 2 sources, avg sentiment 0.00)

Formal notes (optional)

The filter groups rows on the current (possibly transformed) labels using bundle key `(cause label, effect label)`.

In links tables, grouped statistics are attached back to each underlying row. In maps, each group is rendered as one displayed link.

- Bundle key: (cause label, effect label)
- One bundle = one unique cause->effect pair

Bundle-level output fields include:

- `bundle`: a readable key like `cause >> effect`
- `citation_count`: number of underlying link rows in the bundle
- `source_count`: number of distinct sources contributing at least one link row to the bundle

Further summaries can be computed from the underlying rows (e.g. `mean_sentiment`, per-tag counts, per-group counts).

Other extensions like Opposites may themselves extend this extension. For example when combining opposite factor labels you may see additional columns in the links table.

Transformation and interpretation rules

Transformation rule

- **Input:** a links table (one row per coded claim), after upstream transforms.
- **Transformation:** group rows by identical `cause -> effect` labels, then compute bundle-level fields.
- **Output:** a links table with bundle fields (for example `bundle`, `citation_count`, `source_count`) and a map view where each bundle is shown as one displayed link.

Interpretation rule

- A bundle means repeated claims of the same directional relationship.
- `citation_count` is "how often said" and `source_count` is "how widely shared".
- These are evidence-volume measures, not effect-size measures.

The factors table

22 Sep 2025

Summary

The factors table is a **summary view** of your map: it tells you which factors are most prominent in the current view of the data, and how that changes across groups.

- **Input**: the current (already filtered) links table.
- **Output**: the factors table, with one row per factor label, with counts and optional breakdowns.

The key idea: **everything starts from links**. The factors table is created on the fly from the links table. It is not saved separately. In our minimalist causal coding, factors only exist because they are named at each end of causal links.

When to use this table

- **Orientation**: "what are people talking about most?"
- **Role reading**: "which factors mainly show up as outcomes vs causes?"
- **Comparison**: "what differs by group/context?"

What the counts actually mean

The factors table is built from "factor mentions" that come from links:

- each link contributes a **cause mention** and an **effect mention**
- totals therefore count **mentions**, not "number of links"

The main fixed columns in the table are:

- `Citation Count`: total mentions of this factor (`Citation Count: In` + `Citation Count: Out`)
- `Source Count`: distinct sources that mention this factor (as cause or effect)
- `Citation Count: In`: citations where this factor appears as an effect
- `Citation Count: Out`: citations where this factor appears as a cause
- `Outcomeness`: `Citation Count: In / (Citation Count: In + Citation Count: Out)`
- `Influence`: Katz-style cause-side influence score on the directed factor graph (cause -> effect), shifted so minimum is 0
- `Avg Incoming Sentiment`: average `sentiment` of links where this factor is the effect (incoming links only)

- `Source Count: In`: distinct sources where this factor appears as an effect
- `Source Count: Out`: distinct sources where this factor appears as a cause

Two evidence units are used repeatedly:

- **Citations** = how often said (mention/citation volume)
- **Sources** = how widely shared (distinct-source breadth)

Typical views people use

1) Overall prominence

Sort by Source count or Citation count to find the “main” factors in the current view.

2) Causes vs effects

Use these columns:

- `Citation Count: Out` ("as a cause")
- `Citation Count: In` ("as an effect")
- `Outcomeness` (near 1 = mostly outcome; near 0 = mostly cause)

This helps you read whether a factor is mostly described as a driver, an outcome, or both.

3) Group breakdowns (comparisons)

If your sources have metadata (e.g. district, gender, age band), you can break the table down by group to ask:

- “which factors are disproportionately mentioned by group A vs group B?”
- “which outcomes differ by context?”

4) Normalised (percent) views

Normalisation is for fair comparison when groups differ in:

- number of sources, or
- overall verbosity.

In practice: percent views are about **relative prominence**, not absolute volume.

5) Significance tests (optional)

If you choose exactly one grouping variable, the app adds `Significant` (`Yes`/`No`/`N/A`) per factor using a chi-squared-style comparison against group baselines.

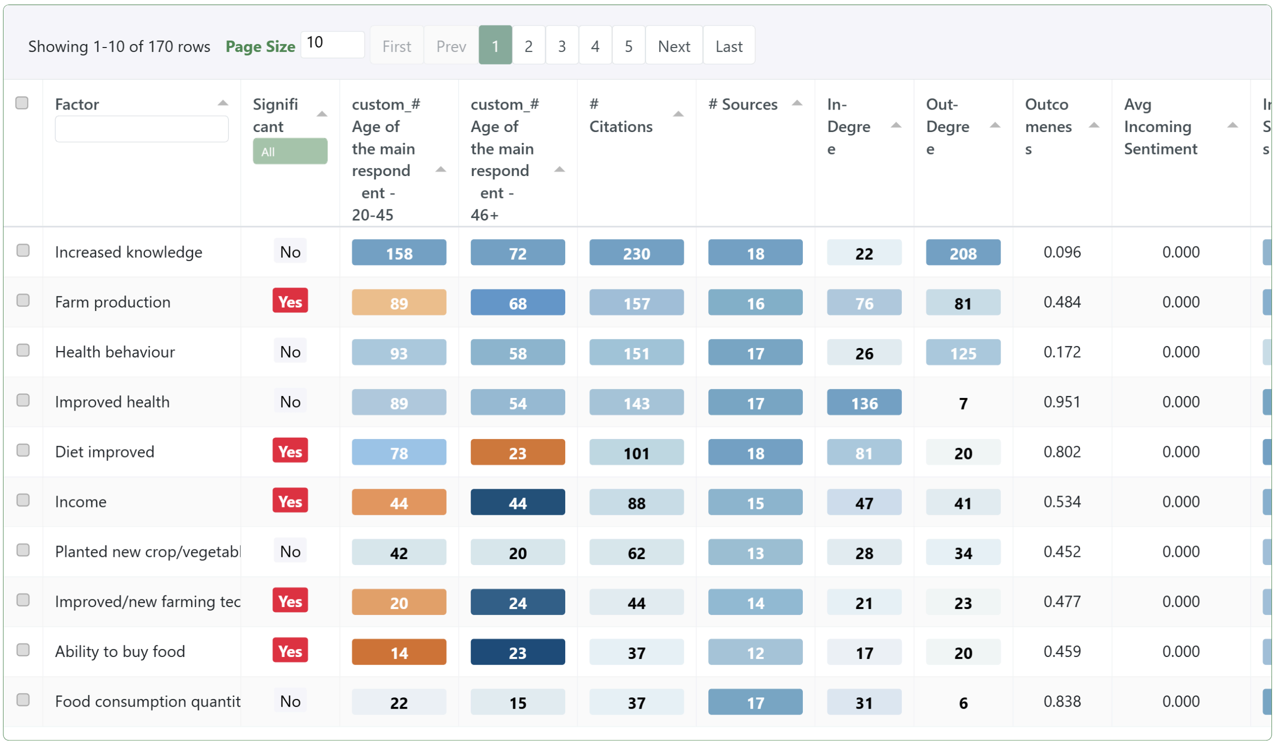
If that grouping variable is numeric-like, the app also adds `Ordinal Sig.` (`Yes`/`No`/`N/A`) from an ordinal trend test.

Use these as **attention guides**, not as definitive proof: always go back to quotes/links to interpret what the difference actually is.

Examples (from the app)

Factors table: group differences + tests

Bookmark [#535](#535)

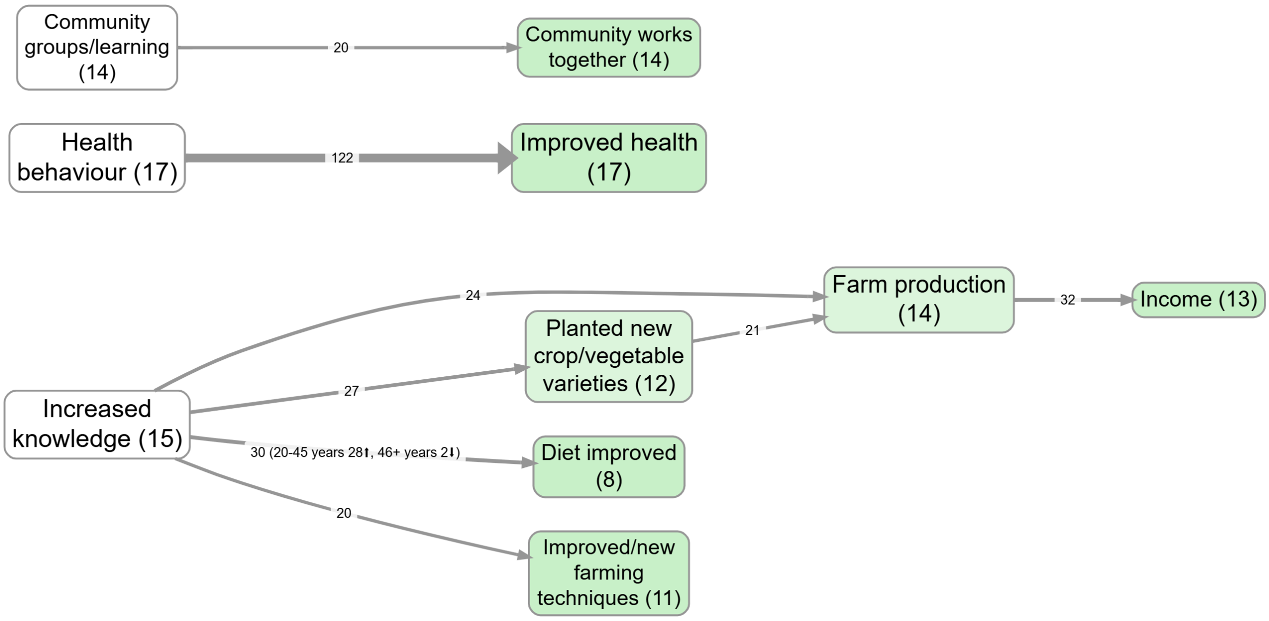


Showing 1-10 of 170 rows **Page Size** 10 First Prev 1 2 3 4 5 Next Last

□	Factor	Significant All	custom_# Age of the main respondent - 20-45	custom_# Age of the main respondent - 46+	# Citations	# Sources	In-Degree	Out-Degree	Outcomeness	Avg Incoming Sentiment	I S s
□	Increased knowledge	No	158	72	230	18	22	208	0.096	0.000	
□	Farm production	Yes	89	68	157	16	76	81	0.484	0.000	
□	Health behaviour	No	93	58	151	17	26	125	0.172	0.000	
□	Improved health	No	89	54	143	17	136	7	0.951	0.000	
□	Diet improved	Yes	78	23	101	18	81	20	0.802	0.000	
□	Income	Yes	44	44	88	15	47	41	0.534	0.000	
□	Planted new crop/vegetabl	No	42	20	62	13	28	34	0.452	0.000	
□	Improved/new farming tec	Yes	20	24	44	14	21	23	0.477	0.000	
□	Ability to buy food	Yes	14	23	37	12	17	20	0.459	0.000	
□	Food consumption quantit	No	22	15	37	17	31	6	0.838	0.000	

Bringing group differences onto the map (as link labels)

Bookmark [#980](#980)


Community groups/learning (14)
20
Community works together (14)
Health behaviour (17)
122
Improved health (17)
24
Farm production (14)
32
Income (13)
Planted new crop/vegetable varieties (12)
21
27
Increased knowledge (15)
30 (20-45 years 28↑, 46+ years 2↓)
Diet improved (8)
20
Improved/new farming techniques (11)

Formal notes (optional)

If you want the precise construction, here it is.

Factor mentions

Each link row contains a cause label, an effect label, and a `source_id`. From each link row we derive two mention records:

- one mention for the cause label (direction = `out`)
- one mention for the effect label (direction = `in`)

These mention records are the atomic units that the factors table aggregates. This is why totals across factors are totals of mentions (each link yields at least two mentions).

Label rewrites

Before aggregating, apply any label-rewrite transforms (collapse, remove bracket text, etc.). These are temporary rewrites for analysis/presentation; they do not change the underlying coding.

Group breakdown cells

If \(G\) is a grouping variable on sources (e.g. district), a cell can be computed in citations-mode or sources-mode:

- in **citation** count type, each mention contributes +1 to its factor/group cell
- in **source** count type, each factor/group cell counts distinct `source_id` values

The dynamic group columns are named `*<group value>` in the table header, one per observed value of the selected grouping variable(s).

Percent-of-baseline intuition

\[\text{share}(f,g) = \frac{\text{cell}(f,g)}{\sum_{f'} \text{cell}(f',g)} \]

Significance tests (intuition)

Even if group A has more mentions overall than group B, the `Significant` test asks whether factor \(f\) is still over-represented in one group relative to those baselines.

Transformation and interpretation rules {.banner}### Transformation rule {.rounded}

- **Input:** a links table (optionally with source-group metadata).
- **Transformation:** derive factor mentions from each link endpoint, aggregate by factor label, and optionally aggregate by selected group values.
- **Output:** a factors table with one row per factor plus counts/ratios (and optional group columns/tests).

Interpretation rule

- Factor-table counts are mention/source summaries derived from links.
- They describe prominence and role (for example cause-side vs effect-side), not causal effect size.

Simplification - factor and link frequency

4 Nov 2025

Summary

This extension is about **simplifying** a causal map by keeping only the **most frequently mentioned**:

- **links** (really: *bundles* of co-terminal links, i.e. repeated claims with the same cause and effect), and/or
- **factors** (the most mentioned concepts/themes).

It is best thought of as:

1. a **filter** (a selection rule applied to derived counts), plus
2. an **interpretation rule** (what “frequency” means and what it does *not* mean).

Core parameters (plain language)

Both the link-frequency and factor-frequency versions share the same conceptual parameters:

- **Mode**:
- **Minimum**: keep everything with count at least the threshold you set
- **Top N**: keep the N highest-count items
- **Count unit**:
- **Citations**: count coded link rows (sensitive to verbose sources)
- **Sources**: count distinct sources (each source contributes at most 1 to an item)
- **Tie rule (important for Top N)**:
- selection typically **respects ties** at the cutoff: if the Nth and (N+1)th items have the same count, you either keep **all** items at that count or **none** (to avoid arbitrary chopping).

Link frequency (keep the most frequent relationships)

What the filter operates on

Strictly, this operates on **link bundles**, not raw individual coded claims.

Start from the current links table (one row per coded claim), then bundle rows that share the same cause and effect labels.

For each bundle we compute at least:

- **citation_count**: number of underlying link rows in the bundle
- **source_count**: number of distinct sources contributing at least one row to the bundle

Then the filter keeps only those bundles meeting the frequency rule (Minimum ≥ k or Top N, using Sources or Citations).

Interpretation rule ("what a link means")

On a map, we often say "link" but we mean:

> the **bundle** representing "many similar claims that one factor influences another".

So "this is a frequent link" means "this cause→effect pairing is frequently claimed", not "the effect size is large".

Factor frequency (keep the most frequent concepts)

What the filter operates on

Factor frequency is derived from the links table by first computing a **factors table** (one row per factor label), with counts such as:

- **citation_count**: how many times the label appears as a cause or effect across coded claims
- **source_count**: how many distinct sources mention the label at least once

The factor-frequency rule (Minimum or Top N, using Sources or Citations) selects a set of "kept" factors.

How this simplifies the map

Once you select the "kept" factors, you simplify the map by keeping only the links whose endpoints are both in that kept set. (In network terms, you are looking at the subgraph induced by the most frequent factors.)

This usually has a nice property: it removes "long tail" concepts while preserving the main structure of the story-space.

What frequency is (and is not)

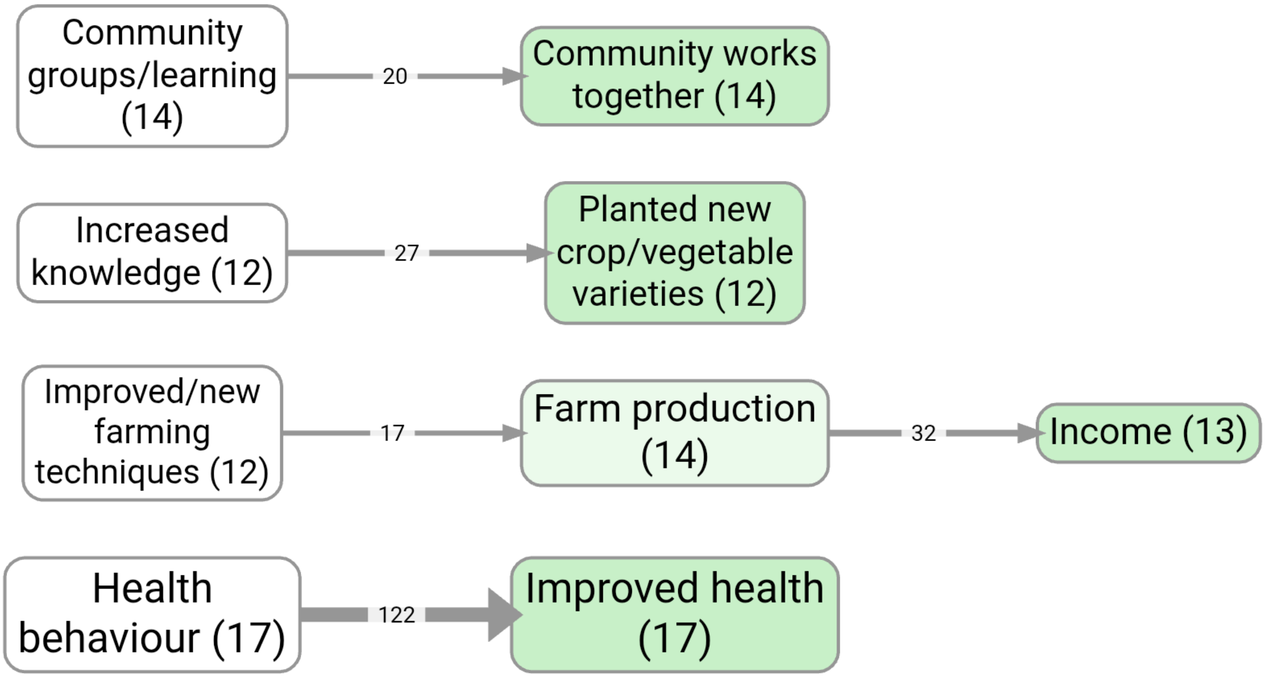
- Frequency is a measure of **evidence volume / breadth**:
- **citations** ≈ how much is said (including repeated mentions)

- **sources** ≈ how widely something is shared across participants/documents
- Frequency is **not** a measure of causal effect size, nor of truth.
- Any frequency-based simplification is therefore a *pragmatic reading strategy*: it prioritises what is most commonly claimed so you can interpret a complex map without being overwhelmed.

Examples (contrasts) from the app

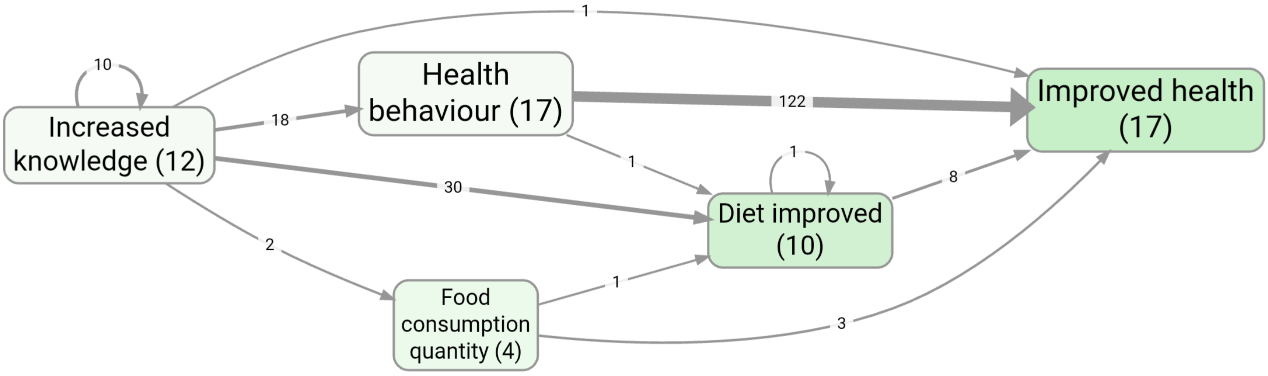
Link frequency (keep the most frequent *relationships*)

Bookmark #1124 shows a "main links" map created by keeping only the most frequent **link bundles** (cause→effect pairs).


Community groups/learning (14)
20
Community works together (14)
Increased knowledge (12)
27
Planted new crop/vegetable varieties (12)
Improved/new farming techniques (12)
17
Farm production (14)
32
Income (13)
Health behaviour (17)
122
Improved health (17)

Factor frequency (keep the most frequent *concepts*)

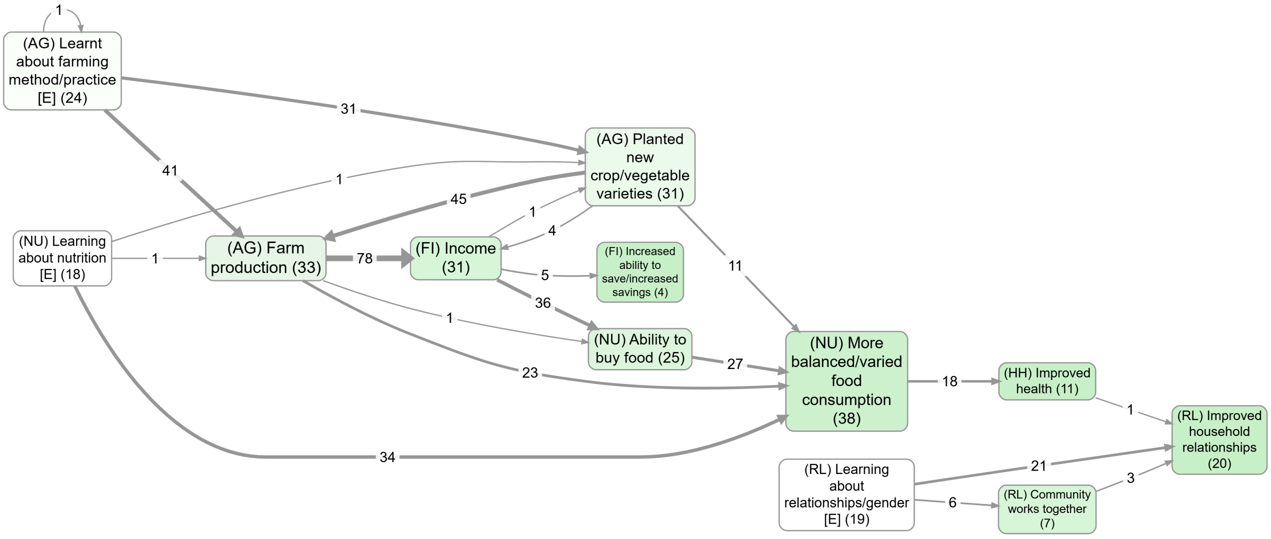
Bookmark #266 shows a "main factors" map created by keeping only the most frequent **factors**.

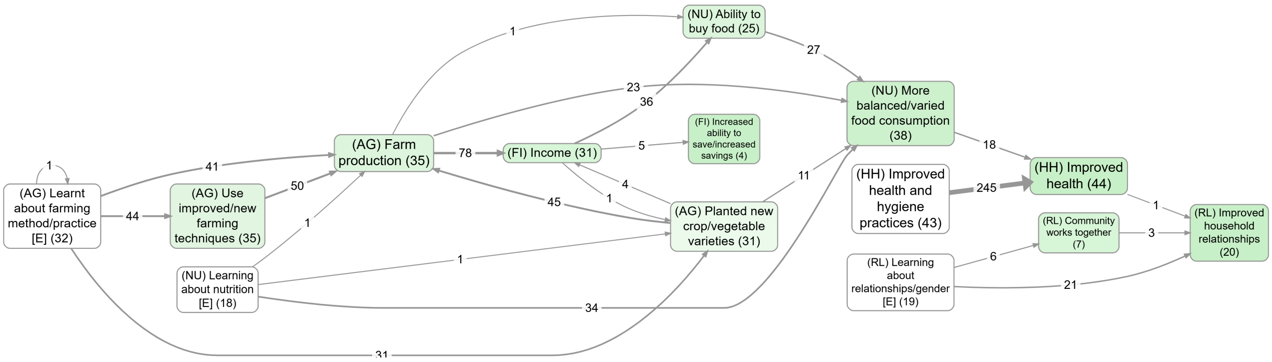


A useful contrast: top factors with vs without zoom

These two views use the same "top factors" idea but differ in **granularity** because zooming rewrites hierarchical labels to a higher level:

- Without zoom: bookmark #983
- With zoom: bookmark #984

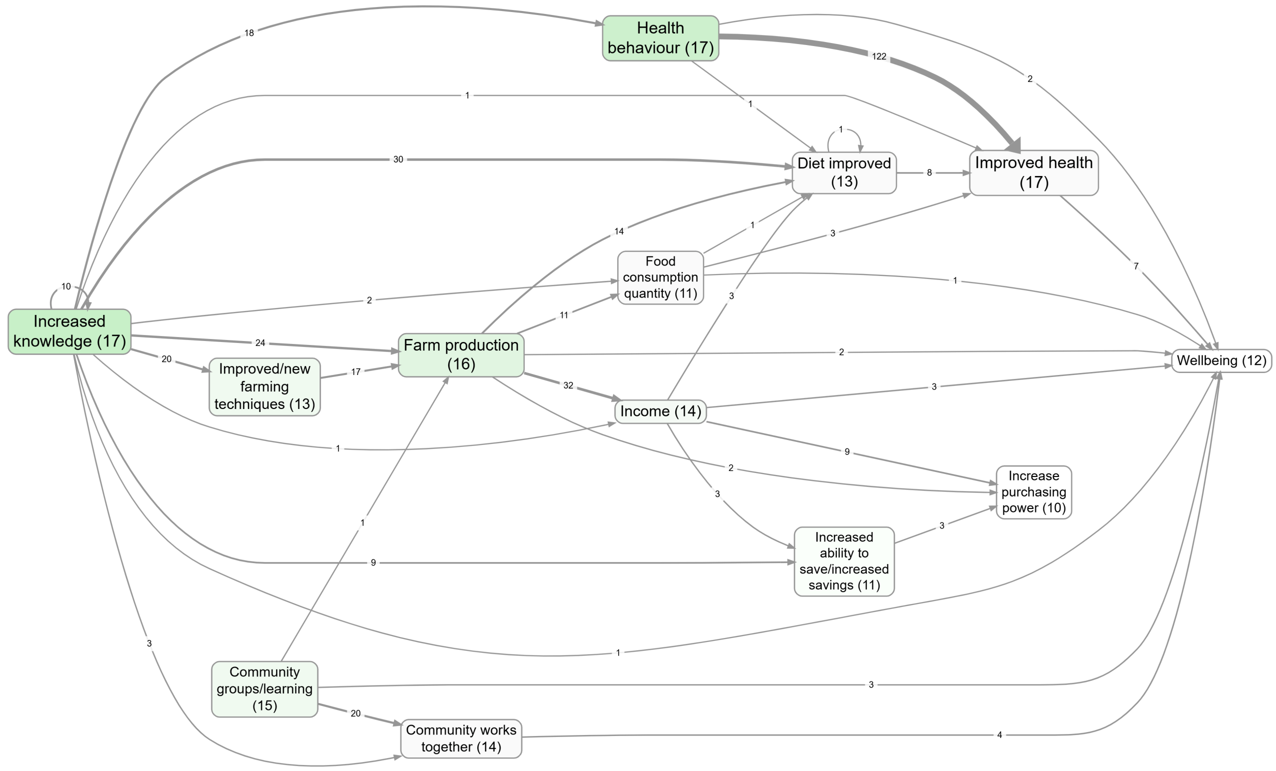

(AG) Learnt about farming method/practice [E] (24)
1
31
41
(AG) Planted new crop/vegetable varieties (31)
1
45
(NU) Learning about nutrition [E] (18)
1
(AG) Farm production (33)
78
(FI) Income (31)
1
4
5
(FI) Increased ability to save/increased savings (4)
36
11
1
(NU) Ability to buy food (25)
27
23
(NU) More balanced/varied food consumption (38)
18
(HH) Improved health (11)
1
(RL) Improved household relationships (20)
34
(RL) Learning about relationships/gender [E] (19)
21
6
(RL) Community works together (7)
3


(NU) Ability to buy food (25)
1
27
23
36
(NU) More balanced/varied food consumption (38)
(FI) Increased ability to save/increased savings (4)
(AG) Farm production (35)
78
(FI) Income (31)
5
18
41
1
(AG) Learnt about farming method/practice [E] (32)
44
(AG) Use improved/new farming techniques (35)
50
1
4
45
1
11
(AG) Planted new crop/vegetable varieties (31)
(HH) Improved health and hygiene practices (43)
245
(HH) Improved health (44)
1
(RL) Community works together (7)
3
(RL) Improved household relationships (20)
(NU) Learning about nutrition [E] (18)
1
34
(RL) Learning about relationships/gender [E] (19)
6
21
31

After simplification: reading "importance" (not just frequency)

Frequency tells you what is mentioned most often. A complementary reading is: which factors are *influential in the network* (e.g. they influence many factors which are themselves influential)?

Bookmark #1063 shows "importance" colouring after simplifying.


Health behaviour (17)
Improved health (17)
Diet improved (13)
Increased knowledge (17)
Food consumption quantity (11)
Farm production (16)
Improved/new farming techniques (13)
Wellbeing (12)
Income (14)
Increase purchasing power (10)
Increased ability to save/increased savings (11)
Community groups/learning (15)
Community works together (14)

Formal notes (optional)

- In Top N mode, selection often respects ties at the cutoff: if the Nth and (N+1)th items have the same count, keep all items at that count (or none), rather than chopping arbitrarily.
- Link-frequency bundling can be defined formally by grouping on (cause label, effect label) and computing `source_count` and `citation_count` per group.

Transformation and interpretation rules

Transformation rule

- **Input:** a links table plus frequency settings (mode: `Minimum` or `Top N`; count unit: `Citations` or `Sources`).
- **Transformation:** either (a) bundle links and rank/filter bundles by frequency, or (b) derive factor frequencies and keep only links between kept factors.
- **Output:** a simplified links table/map with fewer displayed links and/or factors.

Interpretation rule

- Frequency means reported volume or breadth (`Citations` vs `Sources`).
- It helps simplify and orient analysis, but does not measure causal strength or truth.

Exclude links based on group or other metadata

22 Sep 2025

Summary

This extension is about filtering links using **metadata** (information attached to links and/or their sources), not just factor labels.

Typical examples:

- include only links from a particular **district / gender / age band**
- exclude links marked with a **caveat tag** (e.g. `?doubtful`)
- include only links with a particular **sentiment** band or code

When to use it

- **Group comparison**: get a clean view of "group A only" vs "group B only".
- **Quality control**: exclude links you tagged as weak/hypothetical/needs checking.
- **Scoping**: focus on a section of the material (e.g. a particular question, time period, or separator block) if that exists in your metadata.

What it does (plain language)

It takes the current links table (already filtered by your chosen sources and any upstream filters) and keeps/removes rows based on:

- a chosen **field** (a column), and
- one or more **values** you want to include or exclude.

Practical cautions

- **Know what the field means**: some fields live on links (tags, sentiment), others live on sources (custom columns), and some are derived.
- **Order matters**: if you apply transforms that rewrite labels earlier, you may change what you consider "the same" link/factor when comparing groups.

Formal notes (optional)

This is a links-table filter. Given a predicate \(P(\text{link row})\), it returns:

- include-mode: \(L' = { \ell \in L \mid P(\ell) }\)
- exclude-mode: \(L' = { \ell \in L \mid \neg P(\ell) }\)

The predicate is usually defined by: (field = f) AND (value ∈ allowed set), with UI-specific matching rules (exact match vs substring, case rules, etc.).

Transformation and interpretation rules {.banner}### Transformation rule {.rounded}

- **Input:** a links table and a metadata predicate (field + include/exclude values).
- **Transformation:** apply the predicate row-wise to keep or remove matching links.
- **Output:** a filtered links table/map scoped to the chosen metadata condition.

Interpretation rule

- This is a selection/scoping step, not a truth test.
- Result differences usually indicate contextual/group differences in reported claims.

Focus or exclude factors

22 Sep 2025

Summary

This extension is about using **factor labels** to carve out a useful subgraph of your causal map.

Like most extensions, it is best thought of as:

1. **A filter** (a rule that takes one links table and returns another), plus
2. **An interpretation rule** (what it means to say we are “focusing on” or “excluding” factors).

There are two closely related operations:

- **Focus**: keep the causal neighbourhood around one or more “target” factors (their upstream causes and/or downstream consequences).
- **Exclude**: remove unwanted factors (and therefore remove any links that touch them).

Unlike label-rewrite transforms (collapse synonyms, remove bracket text, zoom hierarchies, combine opposites), focusing/excluding does **not** rename factors. It decides which parts of the existing graph you want to *see and analyse*.

How to think about it

Focus = “show me the neighbourhood around this factor”

You choose one or more target factors (by label search), then choose:

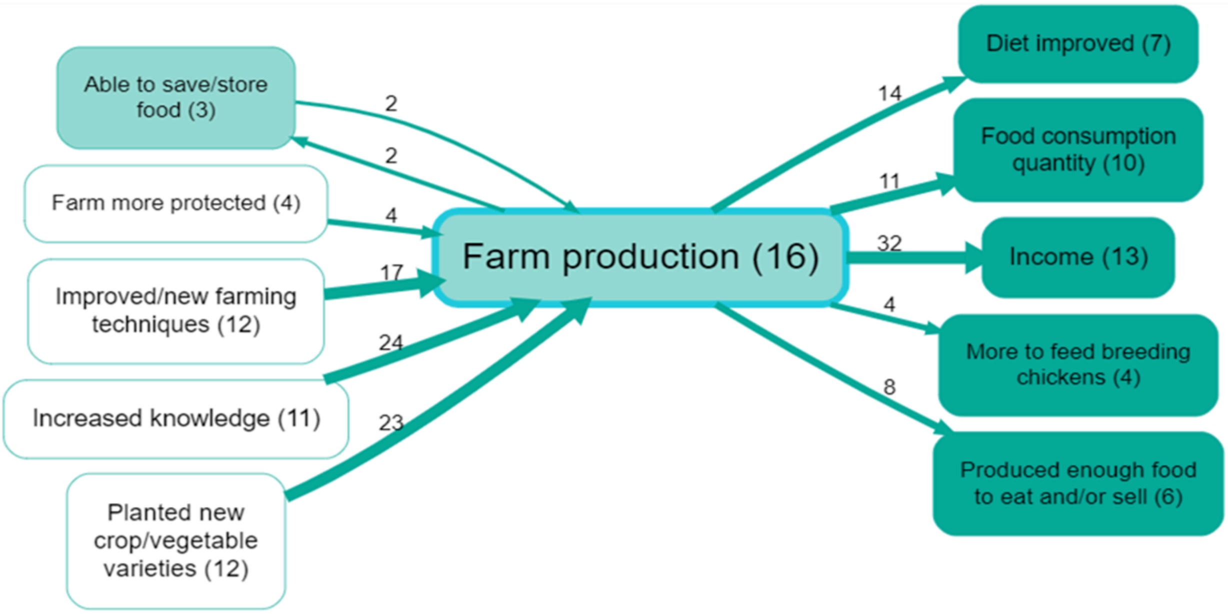
- how far to look **upstream** (causes), and
- how far to look **downstream** (consequences).

The result is a sub-map containing only the links that sit on those upstream/downstream chains.

Focusing is a good way to understand a factor as both:

- an **outcome** (what leads to it?), and
- an **influence** (what follows from it?),

without having to interpret the entire map at once.


Able to save/store food (3)
Farm more protected (4)
Improved/new farming techniques (12)
Increased knowledge (11)
Planted new crop/vegetable varieties (12)
Farm production (16)
Diet improved (7)
Food consumption quantity (10)
Income (13)
More to feed breeding chickens (4)
Produced enough food to eat and/or sell (6)
2
2
4
17
24
23
14
11
32
4
8

Tip: In interview-style data, chains longer than ~4 steps are uncommon. Large step counts can create hard-to-interpret "hairballs".

Source tracing = "only keep paths that appear within a single source"

Sometimes you want coherent within-source narratives rather than a pathway stitched together across respondents.

With **source tracing** on, the focus result becomes more conservative: it keeps only links that lie on at least one upstream/downstream path that can be realised within a single source.

Exclude = "remove these factors (and anything touching them)"

Exclude is subtractive: you specify one or more unwanted factor patterns, and the app removes:

- the matching factors, and
- any links that touch them (as cause or effect).

Interpretation cautions

Order matters

If you apply label-rewrite transforms earlier (collapse, zoom, remove brackets, combine opposites), then focusing/excluding targets are interpreted in terms of the rewritten labels.

Focus is a reading strategy (not a claim about reality)

Focusing is a way to make a large map interpretable. It does not claim “only this neighbourhood is relevant”; it claims “within N steps, what do sources connect to this factor?”

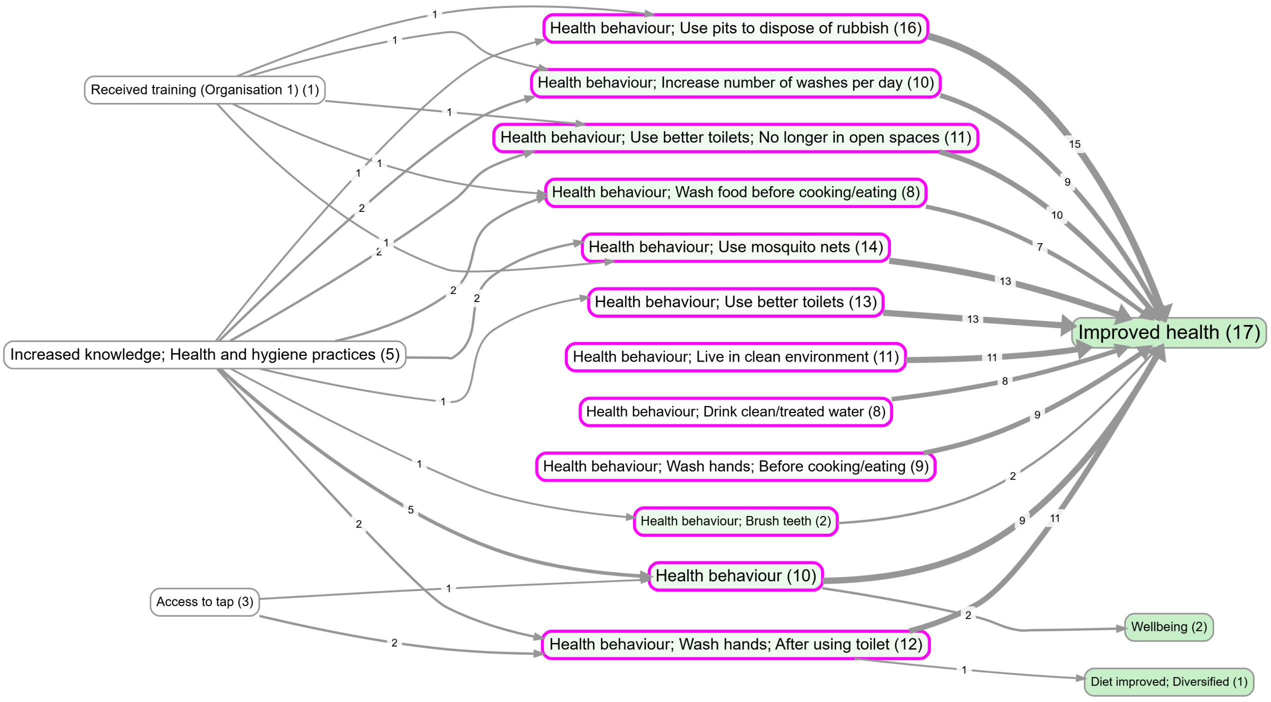
Relationship to “collapse” (different goal)

- Use **collapse/label-rewrite** when you want to treat several labels as *the same concept* while keeping the surrounding structure visible.
- Use **focus** when you want to keep the original labels but restrict attention to the local causal neighbourhood of a concept.

Examples (contrasts) from the app

A single-theme focus (one-step neighbourhood)

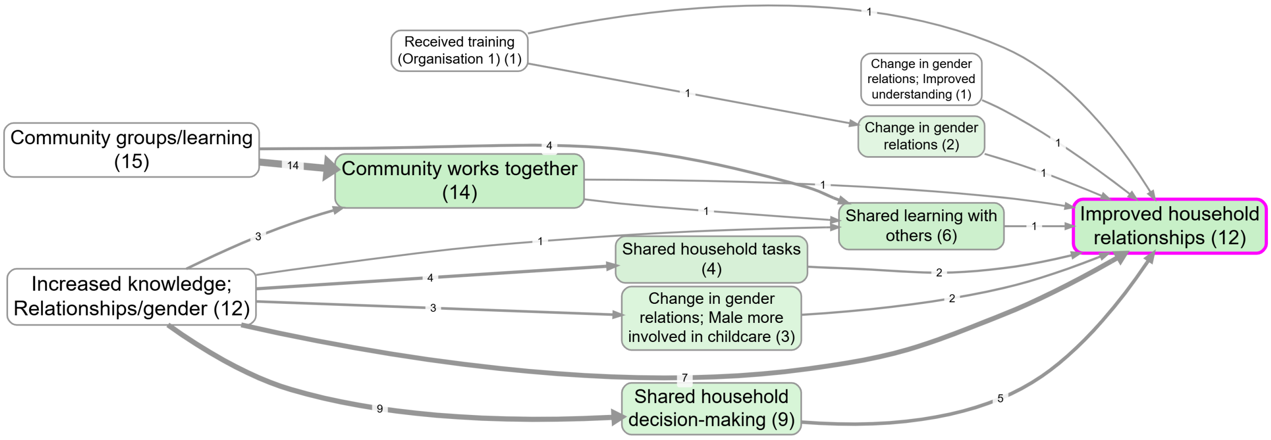
Bookmark [#982](#) is a simple example of focusing on one theme and looking at its immediate neighbourhood.


Health behaviour; Use pits to dispose of rubbish (16)
Received training (Organisation 1) (1)
Health behaviour; Increase number of washes per day (10)
Health behaviour; Use better toilets; No longer in open spaces (11)
Health behaviour; Wash food before cooking/eating (8)
Health behaviour; Use mosquito nets (14)
Health behaviour; Use better toilets (13)
Increased knowledge; Health and hygiene practices (5)
Improved health (17)
Health behaviour; Live in clean environment (11)
Health behaviour; Drink clean/treated water (8)
Health behaviour; Wash hands; Before cooking/eating (9)
Health behaviour; Brush teeth (2)
Health behaviour (10)
Access to tap (3)
Wellbeing (2)
Health behaviour; Wash hands; After using toilet (12)
Diet improved; Diversified (1)

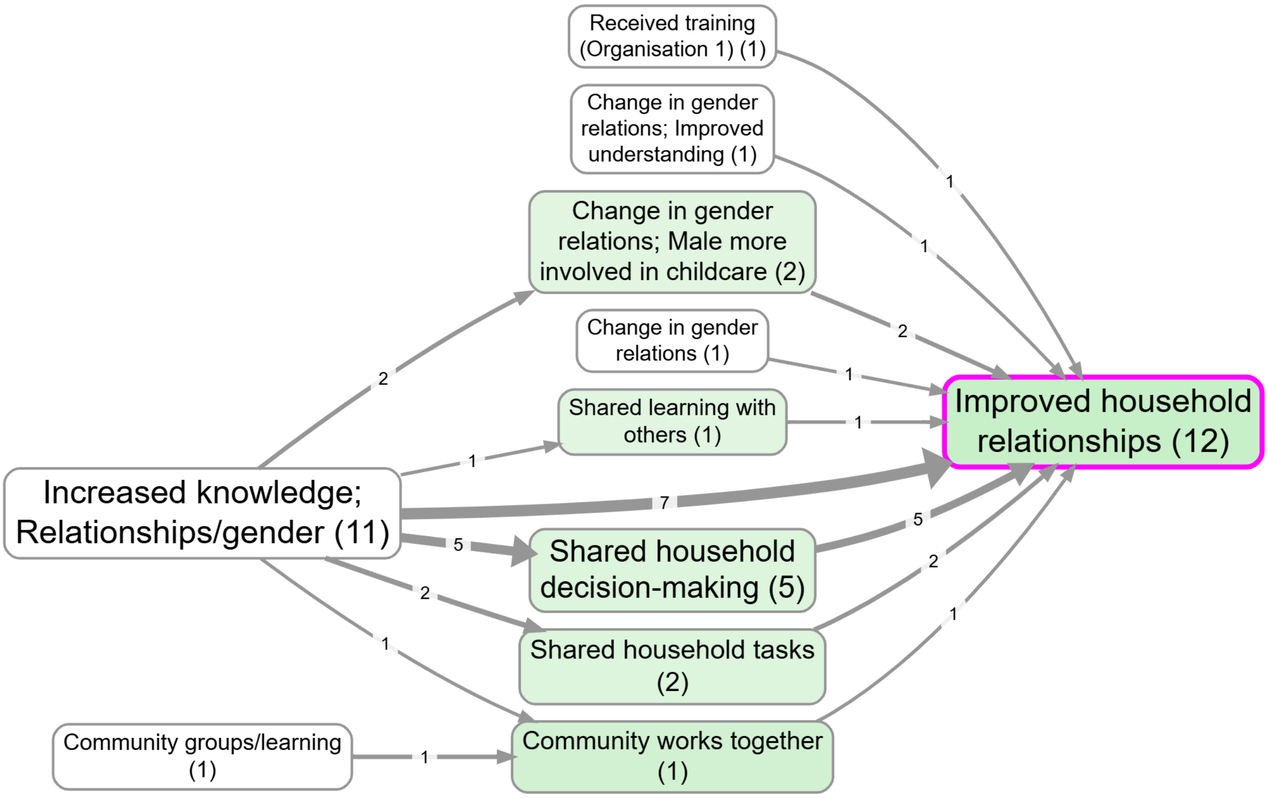
Upstream focus with a single-source constraint (“source tracing”)

These two bookmarks are both “upstream influences on wellbeing” views, but one requires within-source narrative coherence:

- Without source tracing: bookmark [#270](#)


Received training (Organisation 1) (1)
Change in gender relations; Improved understanding (1)
Change in gender relations (2)
Community groups/learning (15)
14
Community works together (14)
4
1
1
Shared learning with others (6)
1
Improved household relationships (12)
1
1
1
1
3
Shared household tasks (4)
Increased knowledge; Relationships/gender (12)
4
2
3
Change in gender relations; Male more involved in childcare (3)
2
7
9
Shared household decision-making (9)
5

- With source tracing: bookmark [#534](#534). Notice how it is more conservative.


Received training (Organisation 1) (1)
Change in gender relations; Improved understanding (1)
1
1
Change in gender relations; Male more involved in childcare (2)
2
Change in gender relations (1)
1
2
Shared learning with others (1)
1
Improved household relationships (12)
1
Increased knowledge; Relationships/gender (11)
7
5
5
Shared household decision-making (5)
2
2
1
Shared household tasks (2)
1
Community groups/learning (1)
1
Community works together (1)

Formal notes (optional)

If you want the precise (link-based) rule, here is the intended definition.

Let \(F\) be the set of focused factor labels, and let \(U\) and \(D\) be the upstream/downstream step limits.

- Keep a link \(x \rightarrow y\) if it lies on any directed path of length \(\le U\) that ends at a factor in \(F\), or any directed path of length \(\le D\) that starts at a factor in \(F\).
- Do not add extra "cross-links" between surviving factors; keep only links that are actually part of the selected paths.

For exclude, let \(E\) be the excluded factor set; remove all links \(x \rightarrow y\) where \(x \in E\) or \(y \in E\).

Transformation and interpretation rules

Transformation rule

- **Input:** a links table plus either (a) focus targets with upstream/downstream step limits, or (b) exclude patterns for factors.
- **Transformation:** keep links on selected focus paths (optionally with source tracing), or remove links that touch excluded factors.
- **Output:** a links table/map showing a focused neighbourhood or a reduced graph without excluded factors.

Interpretation rule

- Focus is a reading strategy for mechanism exploration, not a claim that non-focused parts are unimportant.
- Exclude is an analytical scoping choice; excluded links are omitted for the current view, not invalidated.

Collapsing factor labels and excluding brackets

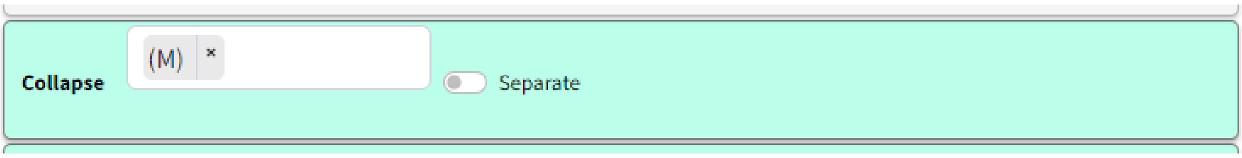
22 Sep 2025

This extension is about **using factor labels to unify many “different-looking” factors into one**.

- **Collapse factor labels**: define one or more search terms, and any factor label that matches is **rewritten to a single shared label** (a “bucket”). This is a quick way to treat near-synonyms and variants as one concept (e.g. *money / income / salary*).
- **Exclude text in brackets** is really the same idea: it’s just a fixed rule for rewriting labels. For example `Education (primary school)` becomes `Education`, so all bracketed variants collapse onto the same base label.

Like the other transform filters, this is a **temporary rewrite** used for analysis and presentation; it doesn’t change your underlying coding.

> In the current app, these text matches are case-insensitive (case doesn’t matter).



Collapse (M) × Separate

All the corresponding factors are collapsed into one factor which is now labelled with the search term.

Multiple search terms, combining results

You can put more than one search term.

Multiple search terms with separate results

Or if you want to keep the results separate. The factors which match the filter are shown with a thicker purple border.

Multiple search terms with OR and separate results

The previous strategy won't work if you want two separate *buckets*, like (Food OR Diet) on the one hand and (Income OR Money) on the other.

In the current app, the clean way to do this is to define separate buckets (so each match collapses to the term it matched), rather than relying on case-sensitivity tricks.

Because matching is case-insensitive in the current app, you don't need separate terms just to capture `Health` vs `health`.

Excluding brackets (same idea, fixed rule)

If you often code specific labels like `Education (primary school)` and `Education (secondary school)`, excluding bracket text is just a convenient way to collapse them both to `Education` without having to maintain a list of search terms.

Transformation and interpretation rules {.banner}### Transformation rule {.rounded}

- **Input:** a links table with factor labels, plus collapse terms and/or the bracket-exclusion option.
- **Transformation:** rewrite matching labels to shared bucket labels (or remove bracketed suffix text), then re-aggregate links/factors on rewritten labels.
- **Output:** a links table/map with transformed labels, fewer distinct factors, and updated counts/bundles.

> **Interpretation rule**

- This is a temporary harmonization of label wording for analysis.
- It combines semantically similar labels but does not change the underlying coded evidence.

Path tracing and source tracing

9 Feb 2026

Summary

Path tracing is for answering questions like:

- “**How does A lead to B** (through what intermediate steps)?”
- “What are the **main routes** from an intervention to an outcome?”
- “If we start from this driver, what downstream consequences show up **within K steps**?”

It’s best thought of as:

1. a **filter** (keep only links that participate in the traced pathways), plus
2. an **interpretation rule** (what counts as evidence for a pathway).

What you get (in plain terms)

You choose:

- a **From** factor (or leave it blank)
- a **To** factor (or leave it blank)
- a maximum number of steps (how long a “chain” you want to allow)

Then the result is a *sub-map* containing only the links that sit on at least one allowed pathway.

When to use it (practitioner-friendly)

- **Explaining mechanisms**: not just “what matters”, but “how it connects”.
- **Finding mediators**: what sits between A and B?
- **Generating hypotheses**: “these are the plausible routes people describe”, then you go back to quotes to check.

Source tracing (recommended when you care about coherent stories)

Without source tracing, a pathway can be a *composite*: one source says A→X and another says X→B, so the map can show A→...→B even if no single respondent told the full chain.

Source tracing is the stricter version: it keeps only pathways that can be realised **within at least one single source** (a coherent within-source narrative).

Use it when you want to avoid the “stitched together across respondents” problem.

Practical tips

- **Keep K small**: in interviews, chains longer than ~4 steps are uncommon and get hard to interpret.
- **Be careful with transforms**: if you change labels (Zoom, Collapse, Combine Opposites, clustering), you can change which pathways exist *as labels*. That’s often fine for presentation, but it changes what you are “counting as the same thing”.

Order matters (a conservative workflow)

If you care about coherent pathways *and* you want a cleaner, summarised map:

- first do **source tracing** (to keep within-source chains)
- then apply label-rewrite transforms (e.g. Zoom / Collapse / Combine Opposites) for presentation

Formal notes (optional)

If you want the precise definition, here it is.

Given one or more start factors \(S\), one or more end factors \(T\), and a maximum path length \(K\):

- keep a link \(x \rightarrow y\) iff it lies on at least one directed path of length \(\le K\) that starts in \(S\) and ends in \(T\)
- if \(S\) is empty, interpret this as “paths that end in \(T\)”
- if \(T\) is empty, interpret this as “paths that start in \(S\)”

The key is: **path tracing is link-based**. It should not “fill in” extra links between surviving factors.

Transformation and interpretation rules {.banner}### Transformation rule {.rounded}

- **Input:** a links table/graph, optional `From` and `To` factors, max path length `K`, and optional source-tracing mode.

- **Transformation:** retain only links that lie on qualifying directed paths; with source tracing, keep only paths realizable within a single source.
- **Output:** a links table/map containing traced pathway links only.

Interpretation rule

- Path tracing shows plausible reported routes under your constraints.
- With source tracing on, routes represent within-source narrative coherence rather than stitched cross-source chains.

Hierarchical coding

22 Sep 2025

Simplifying large causal maps with hierarchical labels (zooming)

Summary

When you code lots of sources, you quickly end up with **too many near-duplicate causes and effects** (and therefore too many nodes on the map). Hierarchical factor labels let you keep the detail *and* produce a smaller “summary map” by “zooming out” to a higher level.

In the Causal Map app we write hierarchies using the separator `;`, but that’s **just one convenient encoding**. The important thing is the meaning: you’re saying “this is a specific case of that, and I'm happy to report it at the higher level if needed".

Example (using `;` to encode "general; specific"):

New intervention; midwife training ➜ Healthy behaviour; hand washing

Read it as:

- "New intervention, and in particular midwife training", or
- "Midwife training, which is part of / an example of the new intervention".

You can extend this to multiple levels:

New intervention; midwife training; hand washing instructions

Crucially, **higher-level labels can also be used directly for coding**. So you can code both detailed links and higher-level links, depending on what the text supports.

What problem this solves (for practitioners)

If you are coding a handful of interviews, you can "just remember" what all your labels mean. But as soon as you code *dozens* or *hundreds* of sources, a common workflow problem appears:

- **Label explosion**: many detailed, overlapping ways to say similar causes/effects.
- **Unreadable maps**: the map becomes hard to navigate and hard to put into a report or slide deck.
- **A trade-off you don't want**: either keep the detail and lose the big picture, or merge too early and lose the details you need for auditability (quotes) and later analysis.

Hierarchical labels are a practical compromise: keep the detailed evidence, but still be able to summarise cleanly.

The idea in one minute

Write detailed factors as:

> General concept **;** specific concept

If you code:

New intervention; midwife training ➜ Healthy behaviour; hand washing

…you can later zoom out and treat this as also supporting:

New intervention ➜ Healthy behaviour

This gives you a smaller map that is easier to read and present, without deleting the detailed evidence underneath.

How to write hierarchical labels (coding convention)

In the app, the convention is:

General concept; specific concept

Two practical conveniences follow immediately:

- Searching for the higher-level label (e.g. `Healthy behaviour`) will also find its nested sublabels.
- Higher-level labels can be created and changed on the fly (they are just the visible prefixes before `;`), without maintaining a separate “parent code” structure.

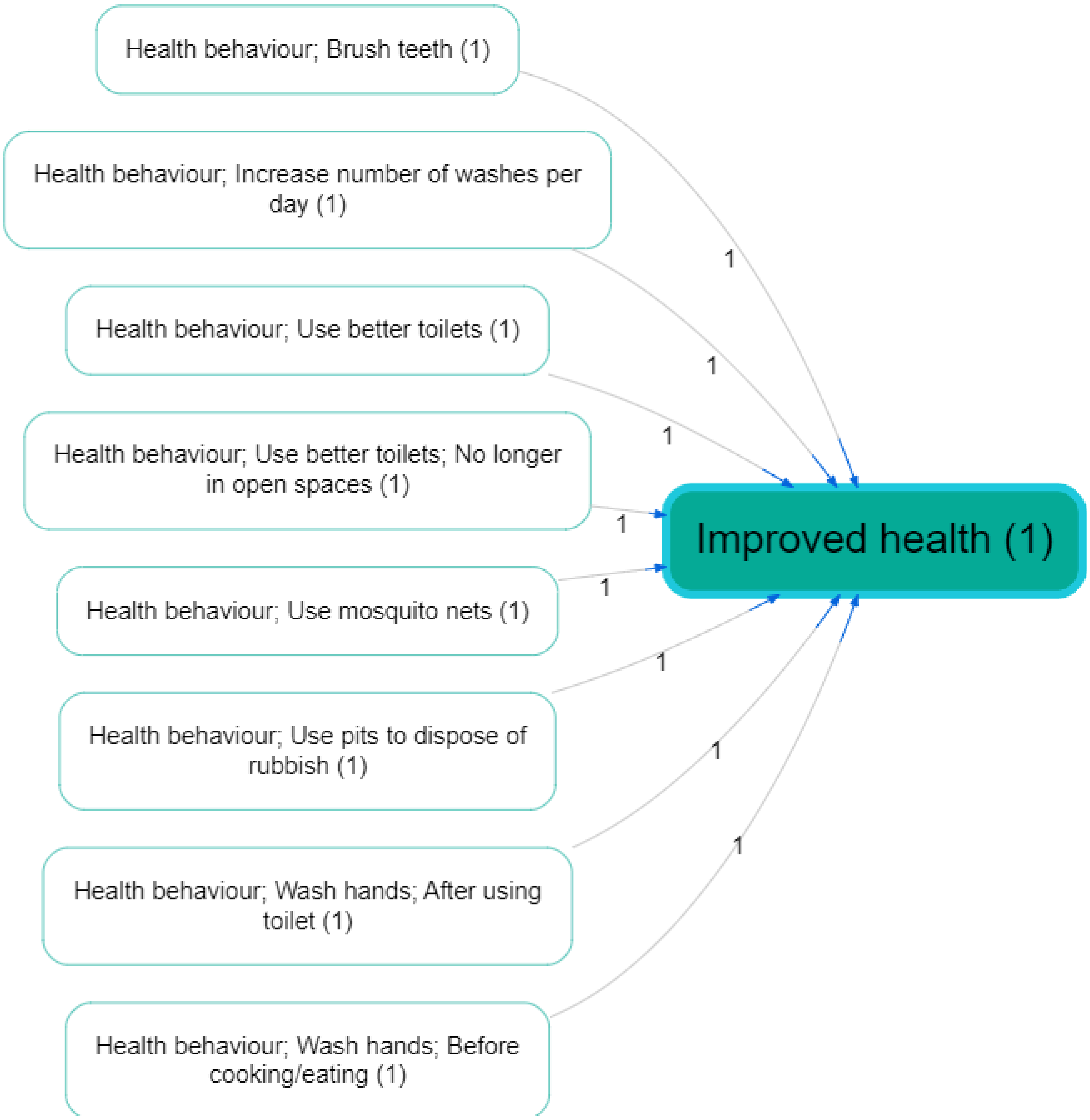
A practical heuristic

When you choose a parent label, pick something you would genuinely be comfortable reporting at a higher level. If “rolling up” the specific item into the parent would mislead a reader, don’t put it in that hierarchy.

AI-assisted coding tip (optional)

If you are using AI to extract candidate links, a simple prompt constraint often helps: ask for factor labels using the template **“general; specific”**, and require a verbatim quote for every link so the evidence remains checkable.

Zooming out (what it does)


Health behaviour; Brush teeth (1)
Health behaviour; Increase number of washes per day (1)
Health behaviour; Use better toilets (1)
Health behaviour; Use better toilets; No longer in open spaces (1)
Health behaviour; Use mosquito nets (1)
Health behaviour; Use pits to dispose of rubbish (1)
Health behaviour; Wash hands; After using toilet (1)
Health behaviour; Wash hands; Before cooking/eating (1)
Improved health (1)
1
1
1
1
1
1
1
1

Assuming you have a causal map that uses hierarchical labels (like the small map shown above), you can "zoom out" by rewriting factor labels to a chosen maximum depth and then redrawing the map.

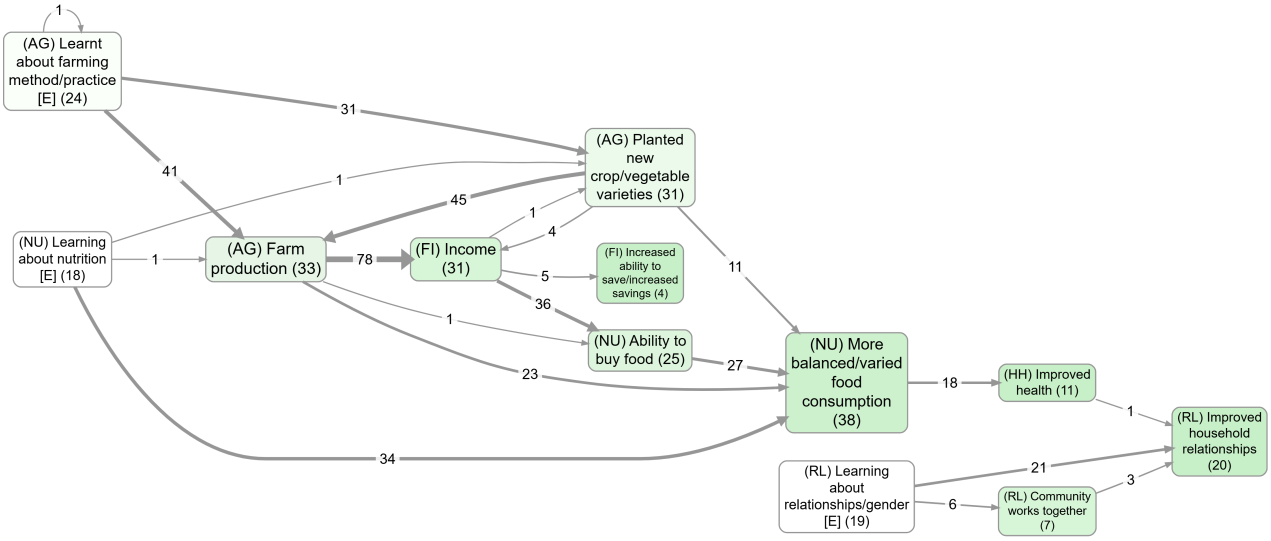
In practice:

- zoom to **level 1** means "keep only the top-level parent"
  Example: `foo; bar; baz` becomes `foo`
- zoom to **level 2** means "keep the first two components"
  Example: `foo; bar; baz` becomes `foo; bar`
- if a label already has fewer components than the chosen level, it stays unchanged.

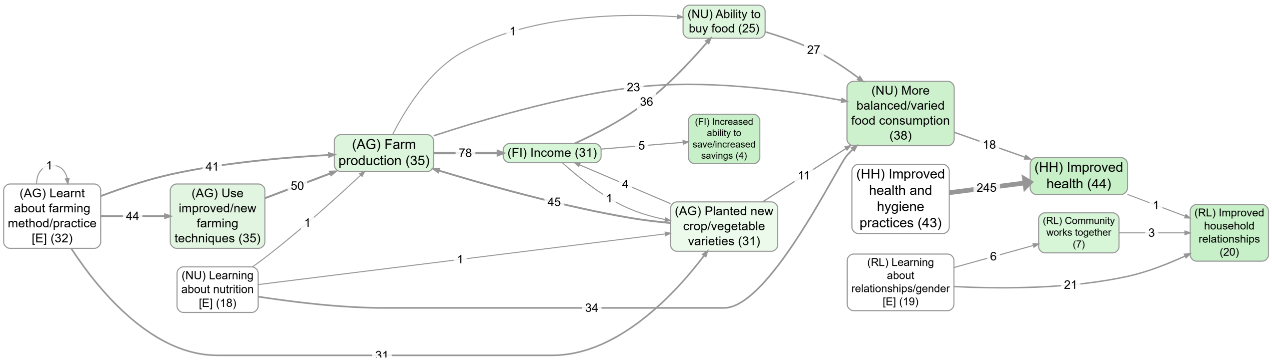
Examples (contrast) from the app

The easiest way to see what zooming is doing is to compare the same "top factors" view with and without zooming:

- Without zoom: bookmark [#983](#983)


(AG) Learnt about farming method/practice [E] (24)
1
31
41
(AG) Planted new crop/vegetable varieties (31)
(NU) Learning about nutrition [E] (18)
1
45
1
4
(AG) Farm production (33)
78
(FI) Income (31)
(FI) Increased ability to save/increased savings (4)
5
11
36
1
(NU) Ability to buy food (25)
23
27
(NU) More balanced/varied food consumption (38)
18
(HH) Improved health (11)
1
(RL) Improved household relationships (20)
34
21
(RL) Learning about relationships/gender [E] (19)
6
(RL) Community works together (7)
3

- With zoom: bookmark [#984](#984)


(NU) Ability to buy food (25)
1
27
23
36
(NU) More balanced/varied food consumption (38)
(FI) Increased ability to save/increased savings (4)
(AG) Farm production (35)
78
(FI) Income (31)
5
18
1
41
(AG) Learnt about farming method/practice [E] (32)
44
(AG) Use improved/new farming techniques (35)
50
4
1
45
11
(HH) Improved health and hygiene practices (43)
245
(HH) Improved health (44)
1
(RL) Community works together (7)
3
(RL) Improved household relationships (20)
1
(AG) Planted new crop/vegetable varieties (31)
(NU) Learning about nutrition [E] (18)
1
34
(RL) Learning about relationships/gender [E] (19)
6
21
31

Don't treat this mechanically: beware the Transitivity Trap

Warning: causal mapping here is a *qualitative* process. Zooming is a powerful summarisation tool, but it only makes sense when your hierarchy choices really do preserve the intended meaning.

Looking at a causal map on its own never lets you safely read off coherent narrative stories along long pathways (the "transitivity trap"). If you want pathway interpretations that reflect single-source stories, use **source tracing** first, then apply zooming/collapsing for presentation.

See:

- Path tracing and source tracing: https://garden.causalmap.app/path-tracing-filter

- Collapse filter: https://garden.causalmap.app/collapse-filter

Zooming out is like a licensed “deduction family”

If you have:

New intervention; midwife training ➜ Healthy behaviour; hand washing

…then (loosely yet informatively, with caveats) you are licensing the summary interpretations:

- New intervention ➜ Healthy behaviour; hand washing
- New intervention; midwife training ➜ Healthy behaviour
- New intervention ➜ Healthy behaviour

This reflects the familiar **granularity** dilemma: how much detail should you code on the cause side and/or effect side? Hierarchies let you keep detail while explicitly stating what higher-level roll-ups are acceptable.

Why this matters in practice

Hierarchical labels + zooming let you do *both* of these:

- **Keep detail**: stay close to what people said and keep your quotes/provenance attached to the specific factors.
- **Communicate the big picture**: produce a readable summary map for slides, reports, decision meetings, and quick orientation.

Typical payoffs:

- **Readable outputs**: a dense map becomes navigable by rolling up to level 1 or 2.
- **Stable high-level story with drill-down**: you can present the summary and still be able to “show your working”.
- **Better search & navigation**: searching the parent label finds its children.
- **Counts at the right level**: you can say “this broad family came up 10 times” while retaining which specific sub-items were mentioned.
- **Scales to many sources (and to AI assistance)**: you can encourage consistent label shape (“general; specific”) while keeping provenance.

Two common scenarios:

- **Many sources, same idea, different wording**: respondents mention “hand washing”, “washing hands before meals”, “soap use”, “hygiene practices”. Hierarchies let you keep those distinctions while still summarising as `Healthy behaviour`.

- **Programme with many activities/outcomes**: training, distribution, mentoring, outreach, etc. Hierarchies let you keep the specific activity while producing a roll-up map for reporting.

Guardrails (useful rules of thumb)

Hierarchies work well *when* the hierarchy is meaningful. These guardrails help you avoid misleading roll-ups:

- **You are licensing a roll-up**: writing `A; B` means you are happy (for summary purposes) to treat this as evidence for `A`. If you wouldn't accept that replacement, don't use a hierarchy there.
- **A factor can't belong to two hierarchies**: you can't have the same specific thing roll up into two different parents at once. If you need cross-cutting grouping, use Factor label tags -- coding factor metadata within its label.
- **Parents should be real causal factors**: higher-level labels should usually be interpretable as causal factors in their own right (often semi-quantitative), not just topics.
- **Don't mix desirability within one hierarchy**: avoid `Stakeholder capacity; Lack of skills`, because zooming out will treat it as evidence for "Stakeholder capacity". Use opposites coding and `~` to fix the polarity, e.g. `~Stakeholder capacity; ~Presence of skills`.
- **Mixed-bag exception**: sometimes you really do want a "bucket" parent (e.g. `Politics; ...`). That's allowed, but interpret zoomed-out links involving that bucket cautiously.

Label factors as events or changes if you can

Hierarchies work best when the parent labels are themselves meaningful causal factors (often semi-quantitative), not just headings or categories. Good examples:

- Experiencing social problems
- Experiencing social problems; Unemployment
- Experiencing social problems; Addiction
- Experiencing psychosocial stressors
- Experiencing psychosocial stressors; Fear of job losses
- Experiencing psychosocial stressors; Pre-existing mental health issues

Here, `Social problems` and `Psychosocial stressors` are higher-level causal factors in their own right; they are not just themes or boxes to put factors into.

So we might have:

> "The problem of unemployment is a psychosocial stress for many"

Experiencing social problems; Unemployment ➜ Experiencing psychosocial stressors

> “When people get stressed they often turn to drugs“

Experiencing psychosocial stressors ➜ Experiencing social problems; Addiction

These could be combined into this story:

Experiencing social problems; Unemployment ➜ Experiencing psychosocial stressors ➜ Experiencing social problems; Addiction

If we zoom out of the above story, we could focus on the higher-level factors and in this case we would get a vicious cycle:

Experiencing social problems ➜ Experiencing psychosocial stressors ➜ Experiencing social problems

When not to use a hierarchy (use a tag instead)

Don’t use `;` to express a non-causal *topic theme*. `A; B` explicitly licenses: “for summary purposes, treat this as `A`”, so `A` must be something you would treat as a causal factor, not just a theme label like “Health”.

For theme-style grouping, use tags (hashtags, brackets, or any consistent convention) inside the label, e.g.:

- Vaccinations law is passed #health
- Mortality rate #health

This keeps the causal claim minimalist (X ➜ Y) while still letting you filter/search by theme.

Hierarchical coding as a way of coping with lots of factors

Analysts are often faced with the quandary of either having too many factors to keep track of, or merging them into a smaller number of factors and losing information. With hierarchies, you can have your cake and eat it: it’s similar to recoding an unwieldy number of factors into a smaller number of less granular items, but with the advantage that the process is reversible. The information can be viewed from the new higher level *and* from the original, more granular level.

For example, you can count that the higher-level component “Healthier behaviour” was mentioned ten times, while retaining the information about the individual mentions of its more granular components.

Re-usable factor components as hashtags

Sometimes your factors relate to each other in ways which are not just hierarchical. For example:

- Activities completed; Training; Health
- Activities completed; Distribution; Health; First-aid kits
- Outcomes achieved; Health; First-aid skills

These are three (hierarchical) factors in which “Health” appears in different places, and at different levels of the hierarchy.

This is not ideal, but sometimes it’s the best way to organise a tricky set of factors.

In this example, “Health” appears only as a component of other factors. Although on its own it might look like a mere theme rather than a causal factor, it plays a role in differentiating the causal factors in which it participates (e.g. “Activities completed, in particular training, in particular on health”). And because “Health” is used across hierarchies, it can *also* be treated as a hashtag and can be used as part of searches, lists and counts of factors.

Isn’t that a contradiction? Didn’t we just say that “Health” is not to be used on its own as a factor because it is just a theme and is not expressed in a semi-quantitative way? No: here the word “Health” is not functioning only as a theme but as a way of differentiating causal factors. The actual factor labels in which it participates are still expressed in a semi-quantitative way. So `Activities completed; Training; Health` is intended as a causal factor about the extent of completion of activities, in particular training activities, and in particular health training activities.

Formal notes (optional)

One clean way to define “zoom level” is by counting components:

- define the level of a label as (number of `;` separators) + 1
- zoom to level N means “truncate the label to its first N components”

See also

- Factor label tags -- coding factor metadata within its label
- Opposites coding

Transformation and interpretation rules

Transformation rule

- **Input:** a links table with hierarchical factor labels (components separated by `;`) and a chosen zoom level.
- **Transformation:** truncate each label to its first `N` components, then recompute aggregates/bundles on the rewritten labels.
- **Output:** a links table/map with transformed labels, typically fewer distinct factors, and updated counts.

Interpretation rule

- Zooming is a controlled roll-up from detailed to higher-level labels.
- Use it when parent labels are valid summaries of child labels; otherwise interpretations can drift.

Opposites

22 Sep 2025

Summary

Opposites coding is for when your data naturally contains paired factors like:

- `employed` and `unemployed`
- `good health` and `poor health`
- `fit` and `not fit`

If you treat those as unrelated labels, you can miss half the evidence when you search/filter/analyse. The goal is to **unify evidence across opposite-coded labels** while still preserving which pole was originally claimed.

Combining opposites

This section presents a simple, practical approach: keep your map “barebones” (plain links between plain-text labels), and use a convention to mark opposites.

When to use it

- When you have meaningful opposite pairs and you want analyses to “see both sides”.
- When you want to compare stories like “X leads to improvement” vs “~X leads to deterioration” without maintaining separate label families.

How to code opposites (two conventions)

1) `~` prefix (simple)

Use `~` at the start of a label to indicate “the opposite pole”:

- `Smoking`
- `~Smoking`

This works with hierarchies too (you can put `~` at the start of a component when needed).

2) Flexible opposite tags

If you want explicit pairing (and sometimes multiple near-synonyms on one pole), use tags like:

- `High income [27]`
- `Low income [~27]`

The exact wording can differ; the number is what binds the pair.

What the Combine Opposites transform does

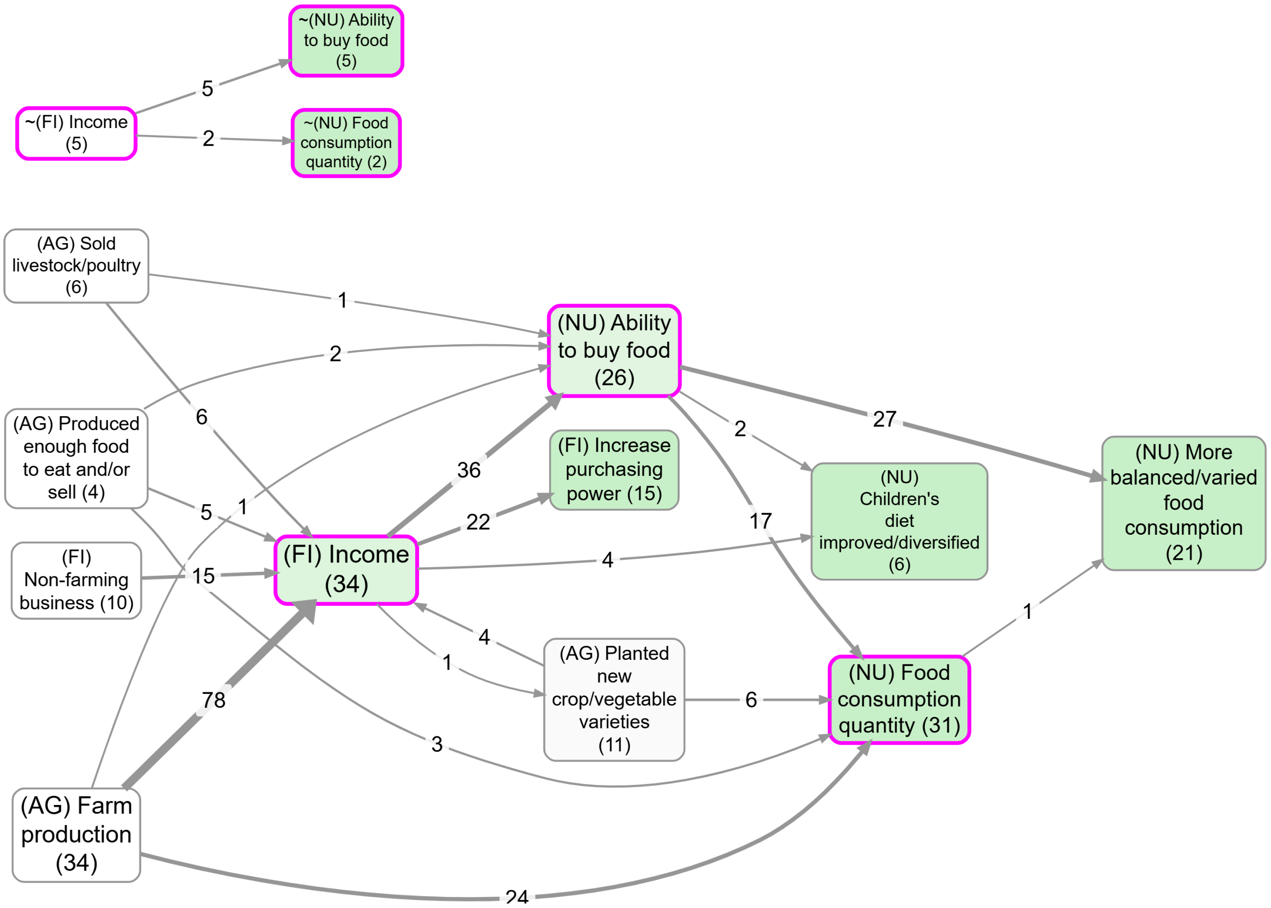
When the app combines opposites, it:

- rewrites opposite-coded labels onto a single “canonical” label, and
- keeps track of which end(s) of each link were flipped, so you can still interpret polarity.

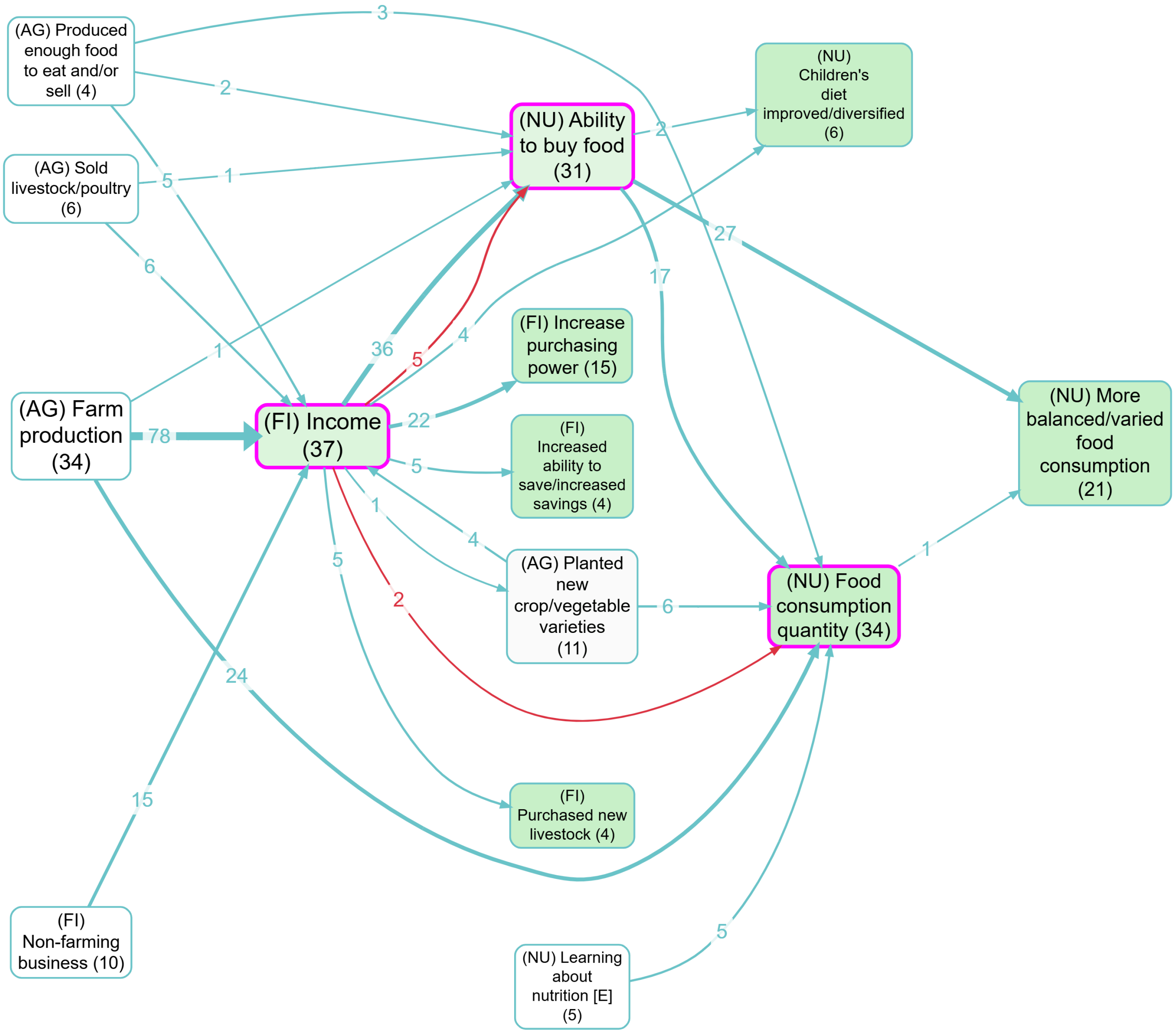
Example (contrast) from the app

The point of “combine opposites” is not to delete anything, but to unify evidence across opposite-coded labels while still keeping track of which pole was originally claimed.

- Without combined opposites: bookmark #985
- With combined opposites: bookmark #986

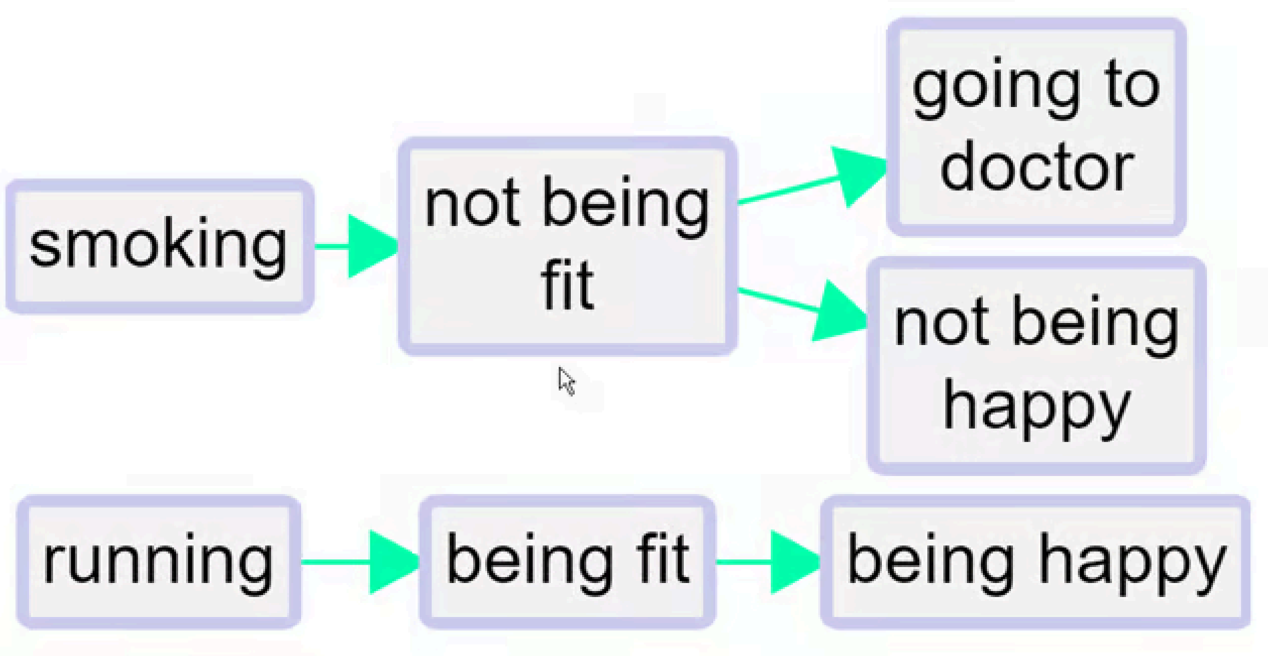


~(FI) Income (5)
~(NU) Ability to buy food (5)
~(NU) Food consumption quantity (2)
5
2
(AG) Sold livestock/poultry (6)
(AG) Produced enough food to eat and/or sell (4)
(FI) Non-farming business (10)
(AG) Farm production (34)
(FI) Income (34)
(NU) Ability to buy food (26)
(FI) Increase purchasing power (15)
(NU) Children's diet improved/diversified (6)
(NU) More balanced/varied food consumption (21)
(AG) Planted new crop/vegetable varieties (11)
(NU) Food consumption quantity (31)
1
2
6
36
22
2
27
17
5
1
15
4
4
1
78
3
6
1
24


(AG) Produced enough food to eat and/or sell (4)
(AG) Sold livestock/poultry (6)
(AG) Farm production (34)
(FI) Non-farming business (10)
(FI) Income (37)
(NU) Ability to buy food (31)
(NU) Children's diet improved/diversified (6)
(FI) Increase purchasing power (15)
(FI) Increased ability to save/increased savings (4)
(AG) Planted new crop/vegetable varieties (11)
(FI) Purchased new livestock (4)
(NU) Learning about nutrition [E] (5)
(NU) Food consumption quantity (34)
(NU) More balanced/varied food consumption (21)
3
2
5
1
6
1
78
36
5
4
2
27
17
22
5
1
4
5
2
6
1
24
15
5

Formal notes (optional)

Here is an example of quite minimalist QuIP-style coding. There are the beginnings of some ideas about (and issues with) polarity: for example, we have `fit`and `not fit`.



We've all done this kind of coding, with classic examples being coding for both employment and unemployment or for both health and illness. This could for example be two different stories about two different people; or it could be different aspects of or periods within one person's life.

This minimalist style on its own is unsatisfactory. We haven't told the app that being fit is represented with a somehow positive and somehow negative factor. So can't join them up. We can't compare the way that being fit leads to happiness and on the other hand not being fit leads to unhappiness (and to visiting the doctor). **We can't for example deduce that running might make visits to the doctor less likely.** Also, if we produce a table or do other analyses focused on healthy habits, we might miss data on the closely related unhealthy habits.

The first step forwards is to follow this convention:

> To signal that two factors are opposites we formalise the idea we already instinctively used in the above example, where we used the word "not" for one of each pair. Formally, we will code them in the form "~Y" and "Y." The ~ may appear at the start of a factor label. This already ensures that when we search for "Y" we will also find "~ Y."In the Edit Multiple Factors panel, these two factors will be listed next to each other - the alphabetical listing will ignore the ~.

We talk about *opposites* rather than positive/negative or plus/minus because that frees us from any implications about valence or sentiment: smoking is the opposite of not smoking, health is the opposite of not health / ill health / illness.

Where there is some kind of valence or sentiment involved, we do suggest using the ~ sign for the negative member of the pair. But it wouldn't make any difference to the app.

So the same map would look like this, using

```
~
```

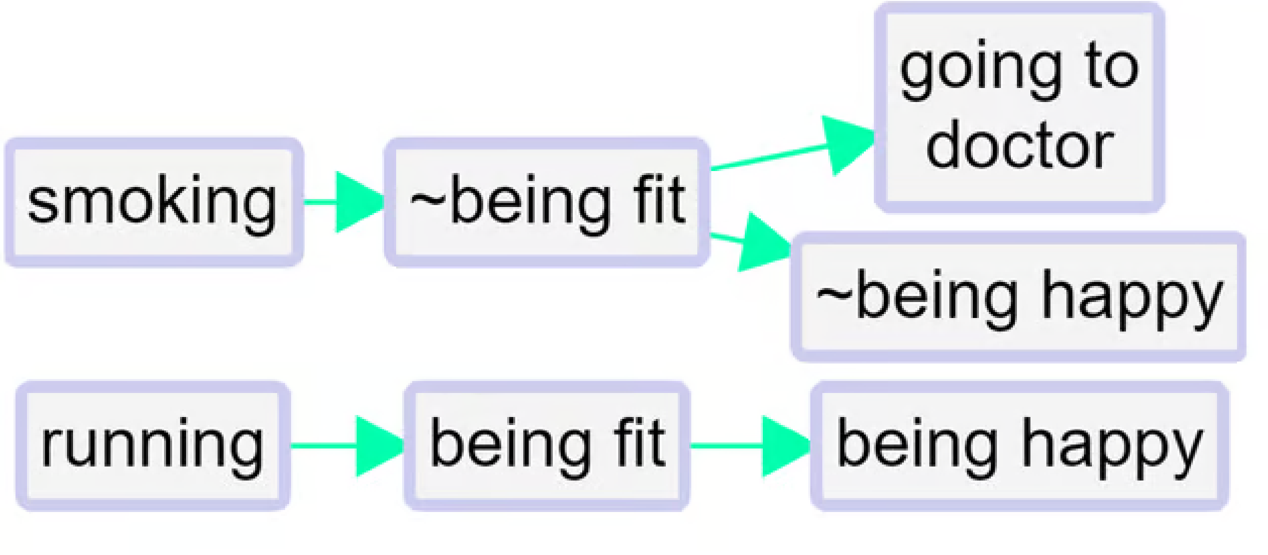
instead of

```
not
```

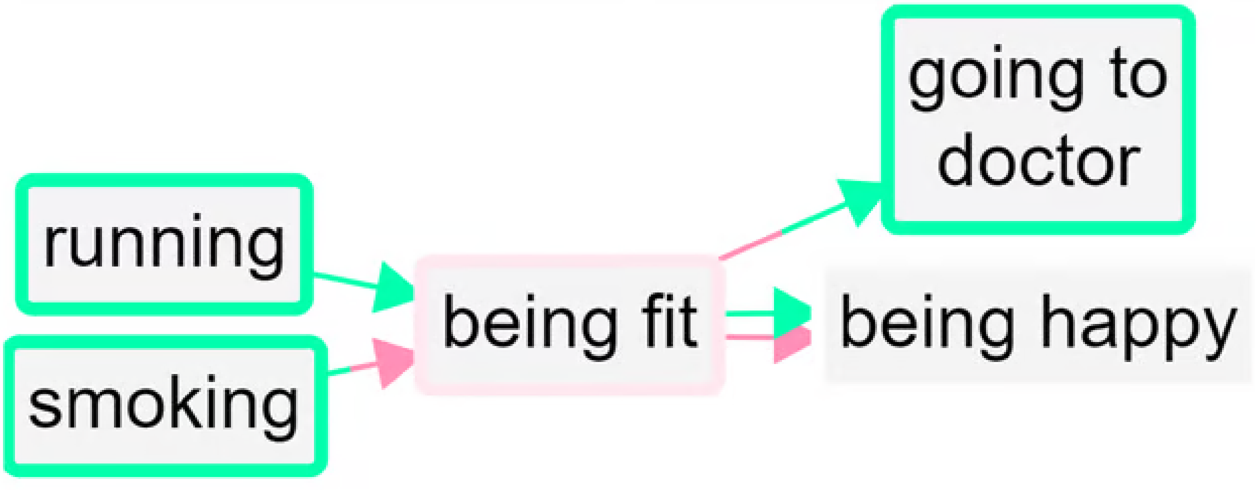
.

Non-hierarchical coding with opposites is easy:

- Eating vegetables
- ~Eating vegetables
- Smoking
- ~Smoking



When you use the “combine opposites” filter (switched on),



the app tries to combine any pairs of factors which are opposites. It looks at all the factors which begin with a ~ and takes off the ~ where there was one. But it only does this if there is in fact such an opposite already coded in the file as currently filtered, otherwise there wouldn't be any point.

So now there are for example two factors combined into the "fit" factor and two into the "happy" factor. The "not" factors have their incoming and outgoing links preserved, but when a factor is flipped to match up with its opposite, the part of the link next to that factor are now coloured pink. So the lower link from fit to happy is pink because the factor at each end of the link has been flipped from "~Y" form to "Y" form; the influence factor was originally *not fit* and the consequence factor was originally *not happy*. So there is no danger of thinking that this is really just another case of the other link, i.e. of fitness leading to happiness.

So, a link has two polarities: a *from* polarity and a *to* polarity. If the signs of the two polarities are opposite, then the effect of the influence factor on the consequence factor is reversed.

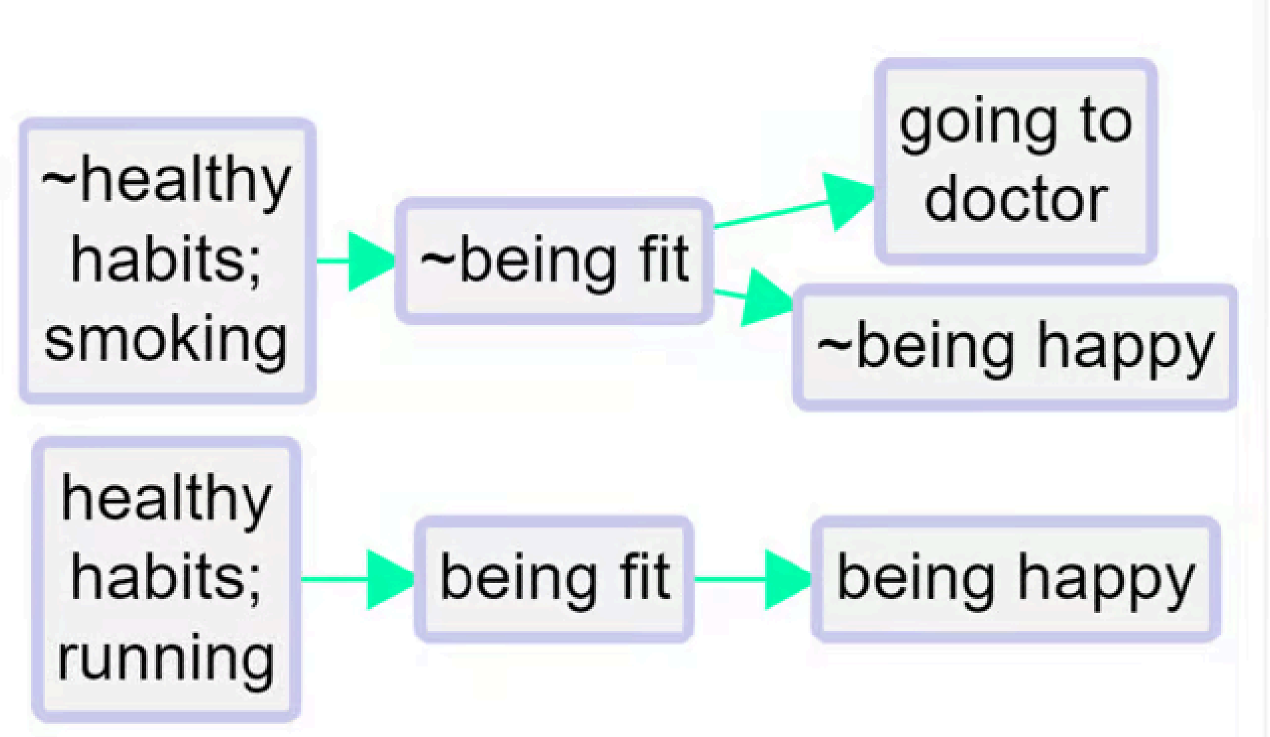
Both links from fit to happy have the same overall polarity (normal, not reversed) but they do not represent the same information.

No information is lost when you press the "combine opposites" button; you can still always read off the original map from it.

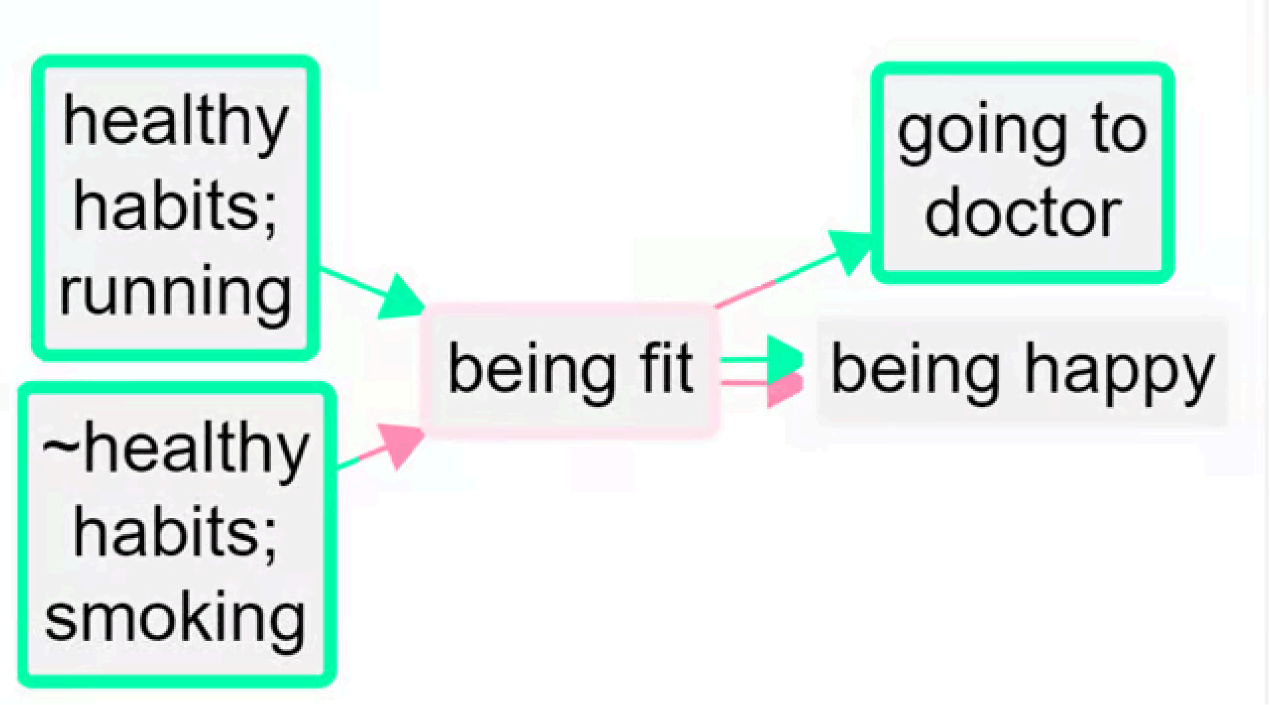
Opposites coding within a hierarchy

When using hierarchical coding, the sign "~" may appear at the start of a factor label *and/or at the start of any component in a factor label.*

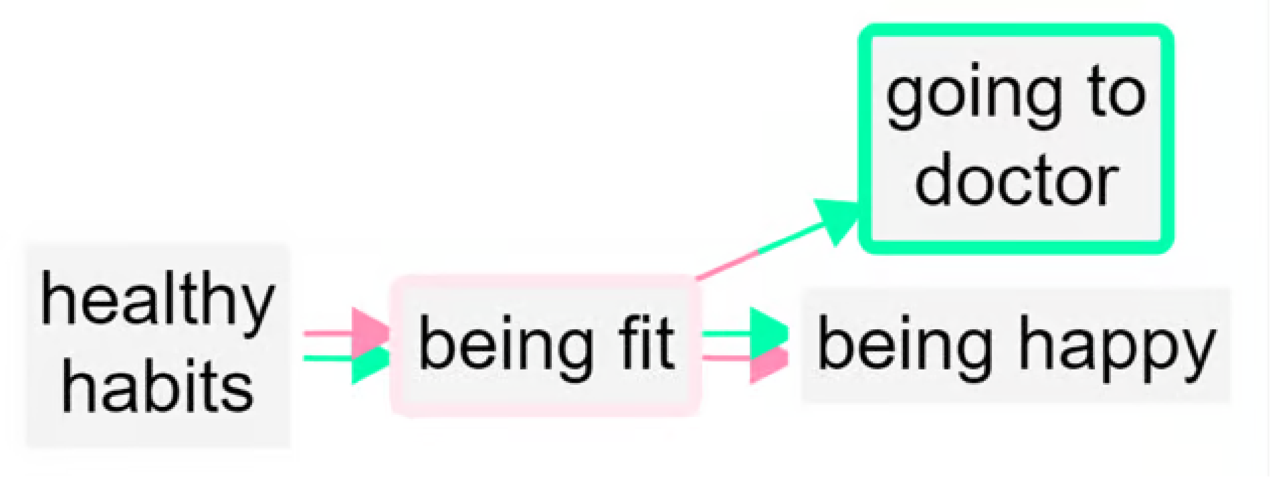
Here is a similar story, now coded hierarchically. In this example, we only see ~ at the beginning of the factors, not yet within them.



When you press “combine opposites,” the app tries to combine any pairs of factors which are opposites. It looks at all the factors which begin with a ~ and *flips each component*, taking off the ~ where there was one, and inserting one where there was not. But again it only does this if there is in fact such an opposite already coded in the file as currently filtered, otherwise there wouldn’t be any point, because there is nothing to combine.



Here is the same example, but also “zoomed out” to the top level.



A quantitative social scientist might solve this problem by flipping the polarity of the negative examples, coding them as positive but using minus strengths for the connections. So smoking influences good health but with a minus strength. However this always seems somehow unsatisfactory and is complicated to do. It is particularly unsatisfactory when *both* ends of the arrow are flipped in this way so that we code the influence of being unfit on being unhappy as a green arrow from fitness to health!

By default in print view, links in the same bundle, i.e. with the same from and two factors, are no longer always displayed as one, with the frequency noted as a label. If we were using the quantitative approach, some of the links in the bundle may have minus rather than plus strength, etc, and we would have to somehow form some kind of average to arrive at an overall strength score, which is not at all satisfactory. Now, the links are only counted together if they have the same *from* and *to* polarities. So there can be up to four different links from one factor to another in Print view.

We are deliberately **not** falling Into the trap of somehow trying to aggregate the different *strengths* to say for example “there are 6 plus links from advocacy to compliance and 1 minus link so this is like 5 plus links because 6 - 1 = 5.” We don’t have evidence for an aggregated strength; we have aggregated evidence for a strength. Aggregating different pieces of *evidence* for links with different strengths is not the same as aggregating links with different strengths. So our more conservative approach preserves information.

It’s also possible that someone says “I know this intervention works not only because the intervention made me happier but also because I saw the people who didn’t get it and they are definitely not happier as a result.” In this case, we might code both arrows, intervention ➜ happy and not intervention ➜ not happy.

Opposites coding *within* components of a hierarchy

Sometimes we need to use the ~ sign *within* the components of a hierarchy.

- ~Healthy habits; ~eating vegetables

is the opposite of

- Healthy habits; eating vegetables

Not eating vegetables, which is an example of unhealthy behaviour, is the opposite of eating vegetables, an example of healthy behaviour.

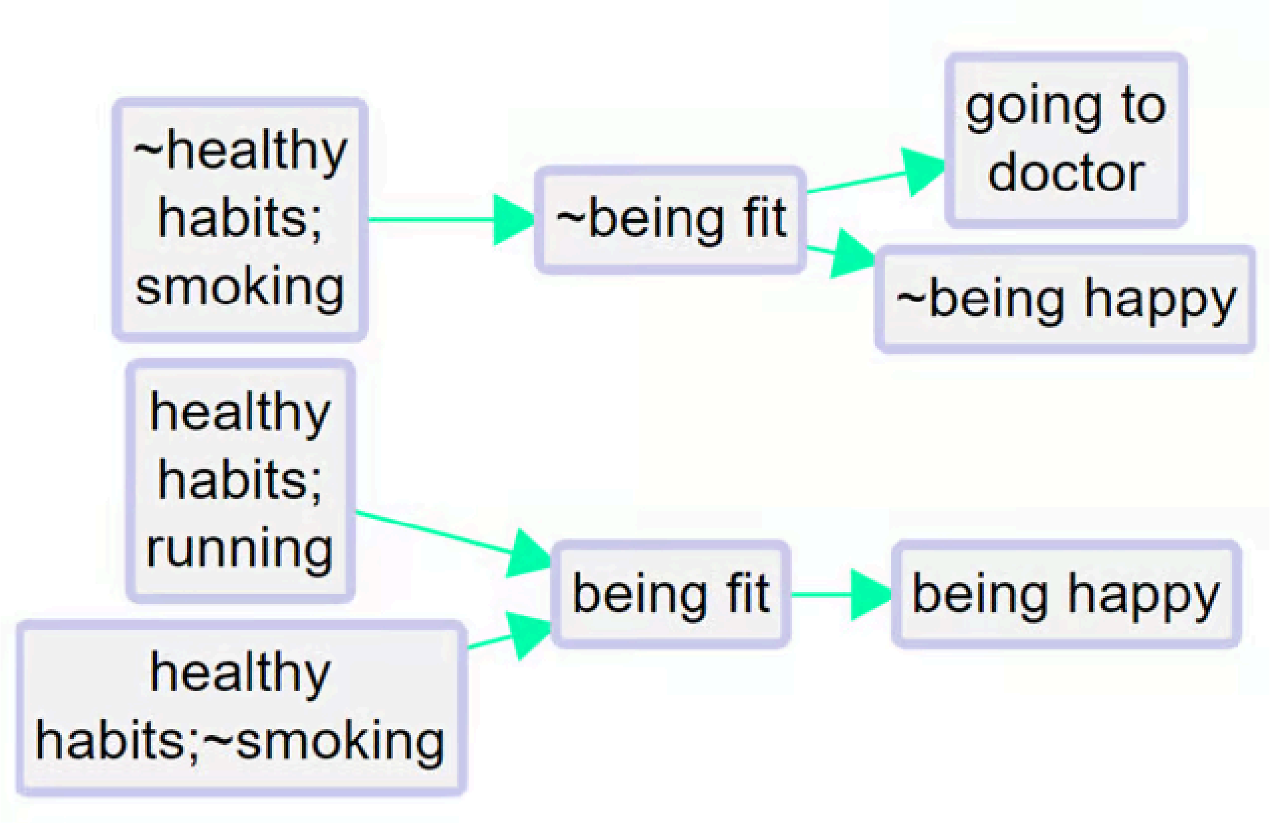
- ~Healthy habits; smoking

is the opposite of

- Healthy habits; ~smoking

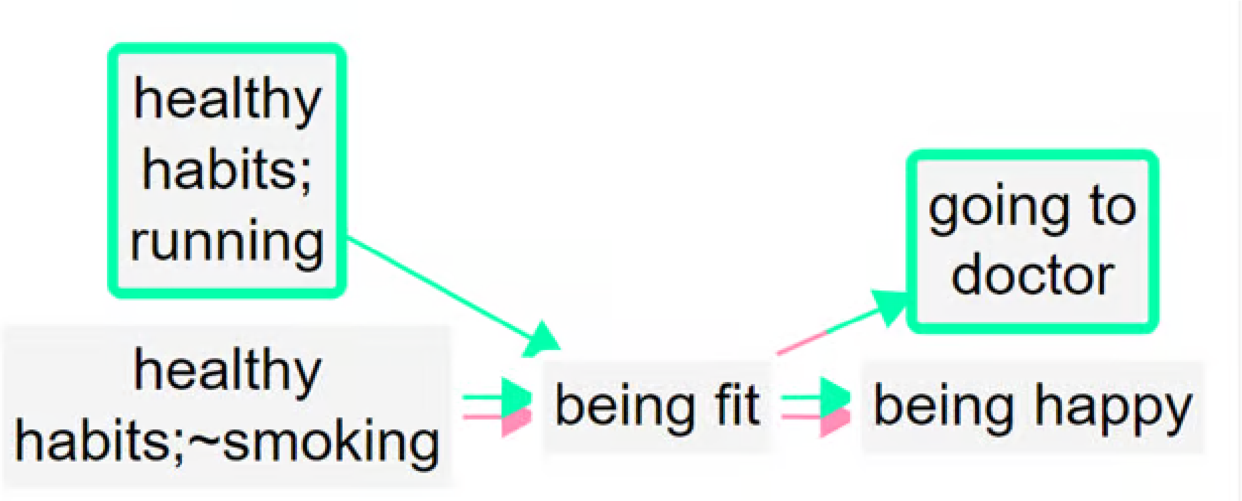
Smoking, which is an example of unhealthy behaviour, is the opposite of not smoking, an example of healthy behaviour.

So here we add one more causal claim to our above example, at the bottom:

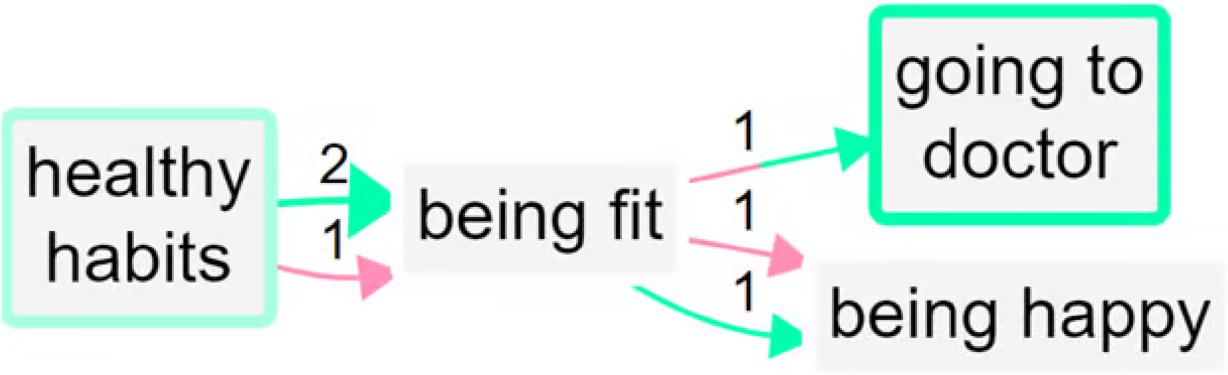


The healthy habit of not smoking leads to being fit.

So the app correctly detects that not smoking is the opposite of smoking:



The two arrows at bottom left (one all green, one all pink) show that there is one example of this particular healthy habit leading to fitness, and the complementary example in which the opposite of this habit leads to the opposite of fitness.



Zoomed out to the top level:

Bivalent variables?

Note that this is the mutually exclusive condition from classical logic, but we don't have any mention of an exhaustive condition: for us, it is not the case that everything has to be either wealthy or poor.

We could imagine that in a causal map, even before we even think about which specific factors might be opposites of other specific factors, each and every factor is really a kind of two-value factor like wealthy (as opposed to non-wealthy) or receiving tuition (as opposed to not receiving tuition). This is the quantitative way of thinking, in which every factor is really a variable which takes at least two values. But this is an unnecessary complication for us. The important point is that usually people don't think of the absence of something as having causal powers (though there are exceptions).

Suppose we are trying to code someone's understanding of income. They tell us that if when people are wealthy, they tend do certain things, and when they are poor they tend to do other things. To some extent, these two sets of things are themselves the reverse or contrary of one another. For example wealthy people will probably have the best education and poor people the least. Sometimes the sets of things we associate with the two opposite poles are not obviously a simple reflection of one another. For example, poor people are often hungry whereas non-poor people are not. But in an affluent country, not being hungry is probably not an important feature we would think of to describe being wealthy as opposed to not-wealthy, if we can assume that non-wealthy people are *overall* rarely hungry. We are more likely to think of things like eating often in posh restaurants.

So being wealthy and being poor are a good example of factors which we *should* consider coding as opposites. We can call the resulting factor, when we press the combine opposites button, *bivalent*.

, we could imagine there was a middle point too:

- Wealthy
- Not-wealthy / Not-poor - a kind of midpoint
- Poor

This is the usual case.

Which pairs of factors should we consider for opposites coding?

Short answer: use opposites coding for a pair of factors X and Y (i.e. recode Y as ~X or recode X as ~Y) if both X and Y naturally occur, separately, in the coding, but they can be considered, broadly speaking, as opposites of one another:

- in particular, it wouldn't normally make sense to apply both of them at the same time in the same situation (you can't be both wealthy and poor in the same sense at the same time)

If in doubt about which of the pair to recode, we usually pick X as the primary member of the pair if it is:

- usually considered as positive / beneficial / valuable
- and/or usually associated with "more" of something rather than "less" of something.

Transformation and interpretation rules

Transformation rule

- **Input:** a links table with factor labels which use opposite coding convention (for example `Wealth` and `~Wealth`, or paired tags like `Wealth [27]` and `Poverty [~27]`).
- **Transformation:** rewrite opposite-coded labels to one canonical label (just `Wealth`) while storing, in the links pointing to or from these "flipped" labels, the information that the factor has been "flipped".
- **Output:** a links table with unified bundles/rows that still preserve direction and polarity metadata.

Interpretation rule

- Combining opposites merges evidence across poles for analysis convenience.
- It does not erase original meaning; polarity still matters for interpreting link direction/sign.

Plain coding

22 Sep 2025

Summary

Causal mapping doesn't usually deal with the kind of non-causal themes which are the focus of ordinary QDA (like in NVivo!). However sometimes it can be really useful to be able to simply note the presence of something without any causal connection.

We call this "plain coding". You can use it for:

1. Noting the presence of something which is not mentioned as part of a causal link in the statement you are coding but does appear *elsewhere* as a causal factor as part of a causal link.
2. ...Or noting the presence of something which is "nothing but" a theme and never appears in causal coding.

Causal Map makes this possible in a simple way: we define a "plain coding" as simply a link from a factor to itself with has the hashtag `#plain_coding`.

Factors involved in plain codings (whether some of the time or all of the time) can of course still be involved in hierarchies and opposites coding just like any other factor.

So by default, plain codings will still appear in maps, as self-loops from the factor to itself, although it is possible to exclude such links by using the appropriate transforms filter.

Doing plain coding like this has the big advantage that factors coded with plain codings will still appear in the tables in the ordinary way, so for example the plain coding to (and from) the factor increased income will still count towards its source and citation counts.

It also means that you can easily delete a plain coding from a map by clicking on the link.

The hashtag `#plain_coding` distinguishes plain codings from actual causal links where a factor indeed influences itself, for example if someone says that increased income led to even more increased income – in this case the hashtag is not used. This hashtag can also be used together with other hashtags for the same link in the usual way.

How the Causal Map app implements this:

See the in-app help section on the Link Editor (Plain coding): https://app.causalmap.app/help-docs#causal-overlay

Transformation and interpretation rules

Transformation rule

- **Input:** a factor mention you want to record without asserting a causal relation.
- **Transformation:** encode it as a self-link (`factor -> factor`) tagged `#plain_coding`.
- **Output:** a links table/map entry that is filterable/countable but distinguishable from ordinary causal links.

Interpretation rule

- Plain coding marks presence/theme evidence, not mechanism.
- Keep it separate from real self-causation claims by using the `#plain_coding` tag.

AI extensions – Clustering, Soft Recode, Magnetisation

22 Sep 2025

Why magnetic labels

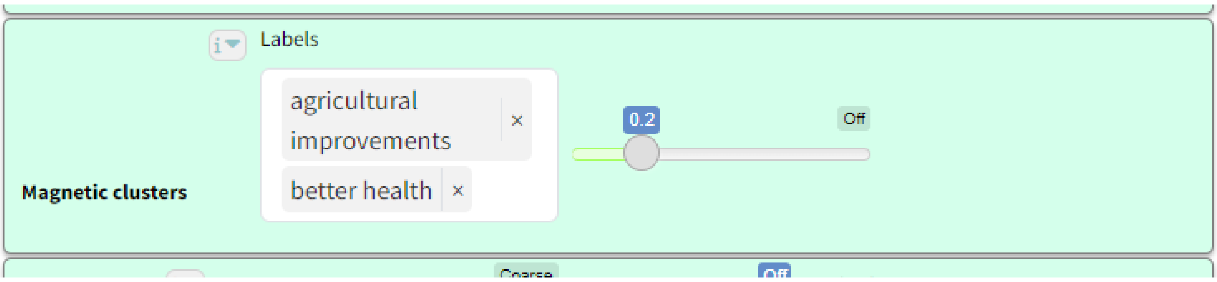
You have already coded your dataset, manually or using AI, and now you want to relabel.

Suppose you already know what labels you want to use, perhaps:

- you knew before you even started
- you decided what labels you wanted after reviewing your data and looking at different auto-cluster solutions

Magnetic labels are a really simple solution for these cases.

How to use them



Magnetic clusters

Labels

agricultural improvements ×

better health ×

0.2

Off

You simply type the list of magnetic labels you want and decide on the power of the magnets (”magnetism”).

Magnetic labels attract existing labels of similar meaning, essentially relabelling these old labels with the new magnetic label. If an existing label is similar to two or more different labels, it is relabelled with the magnetic label it is most similar to.

If you use low magnetism, the magnets are weak and only attract existing labels which are very similar to them.

Increasing the magnetism means that more and more existing labels are attracted to the magnetic labels.

Existing labels which are not attracted to any label are unchanged. This means that you can easily see if your magnetic labels cover most of the original content.

Best practice is then, after applying magnetic labels, to then auto-cluster the links in order to pick out important themes which are not covered by the magnetic labels.

> If you want you can even include hierarchical magnetic labels like `Health behaviour; hand washing`.

Storing your labels

Your magnetic labels are included if you save a bookmark aka saved View.

You can also store your preferred label set in the Codebook in The Files tab.

Use cases

- Drop in magnetic labels which contain the text from the “official” theory of change.
- See how much the existing labels get attracted to the magnetic labels, and what material is left over.
- Conduct “radical zero shot” auto coding with no codebook at all,
    - let the AI decide the best label for each case
    - do some auto clustering until you get a feel for the labels you really want in your story
    - type the labels into the magnetic clusters box

Tips

What if you have a *semantically ambiguous label?*

An example: you have some research about animals and you want to look for mentions of the organisation Animal Aid. If you use Animal Aid as a label, it might also pick up any mention of helping animals which have nothing to do with the organisation itself.

One way to get round this is to use 🔗 The Manage Links tab to permanently recode any mention of Animal Aid in your factor labels into something unambiguous like, say, The Archibald Organisation. Choose a meaningless name which is not going to appear in or be related to to the rest of your material.

When doing this “hard recoding” remember to recode AnimalAid, Animals Aid etc as well.

Attracting unwanted material *away* from your map

You can add an factor into magnetic clusters even if it doesn't appear in the final map.

For example you might have a lot of material about blood donors and you don’t want material about donating clothes. As well as donating blood you might add the labels donating goods and donating money. You can filter these out later, but they will help restrict donating blood to what you want.

Increasing coverage with hierarchical magnetic labels

It might just be that your interview material is so heterogeneous that, however you choose your magnetic labels, if you only have say 10 or 20 of them then they are just not going to cover more than say 30% of your links in all of your stories and that's just the way it is because the material is very broad.

You might have hoped to arrive at a kind of global mind map - but the best you can do is just these most frequently mentioned common factors. You'd have to accept that it isn't in any sense a summary of *all* of the material because there's lots of other stuff that doesn't feature amongst the top 10 or 20 magnetic labels.

You might then also want to focus on more specific maps for more specific subject areas.

Or maybe you have a sense that in fact much of the material really is held in common but you're struggling to find the right magnetic labels? One way to increase coverage is to use hierarchical magnetic labels, of which you might have even 30 or 60 or even 100, and then zoom out to level one. So you might have, say, magnets like:

Desire for innovation; digital

Desire for innovation; management approaches

....

And then you'd apply a zoom level of 1 in order to bundle these things together.

Transformation and interpretation rules {.banner}### Transformation rule {.rounded}

- **Input:** a links table with existing factor labels, a user-provided list of magnetic labels, and a magnetism threshold.
- **Transformation:** for each existing label, map it to the most similar magnetic label if similarity is high enough; otherwise keep the original label unchanged.
- **Output:** a links table/map with partially transformed labels, updated bundles/counts, and visible uncovered material.

Interpretation rule

- Magnetic labels are a soft recoding layer for harmonization and exploration.
- Stronger magnetism usually increases coverage but can also pull in weaker semantic matches.

Social Network Analysis – SNA

For this extension, we can simply tweak our coding rule to code sending/receiving relationships between social actors instead of causal links between factors or events.

Social Network Analysis (SNA) is a useful tool for understanding and visualising connections between actors.

- In a **causal map**, the links are of just one type: X causes Y.
- A **social network map** is similar but the nodes are **actors** and the links can be of different types like "supports" or "knows" and they can have a direction or be undirected, like "works with".

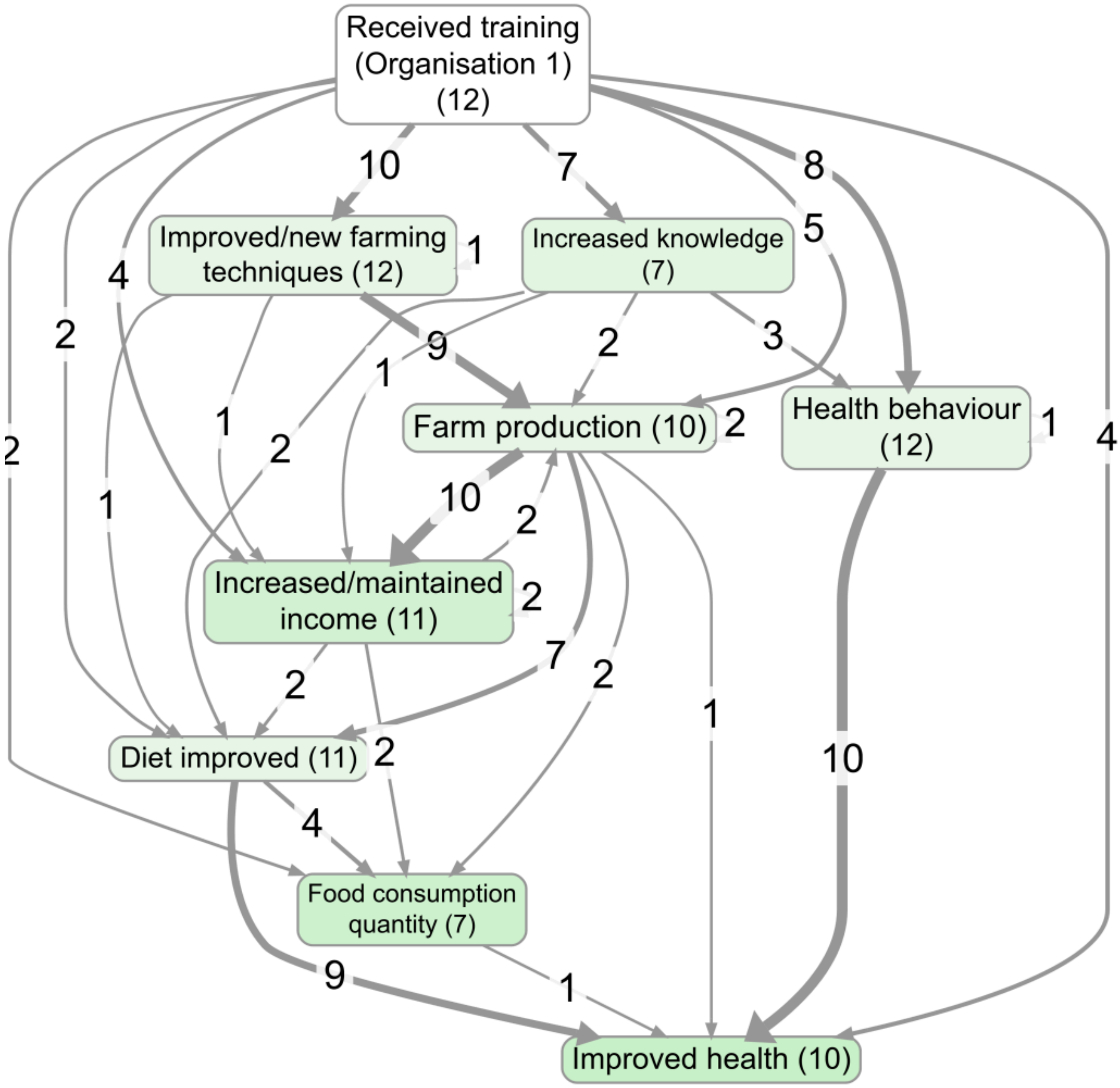
But how do you actually draw a social map?

1. With SNA just as with causal mapping you are faced with a choice. You can, just like in participatory systems mapping, simply get some experts or stakeholders to sit in a room and **collaboratively construct a map** -- out of their heads, so to speak -- with a whiteboard or some software like sticky.studio or prsm.uk, and that can work really well.
2. But what if you want to do it more systematically and **empirically on the basis of, say, a whole bunch of interviews or reports**? It can be quite wearisome -- just like manual causal mapping at scale.

You can do it quite simply with Causal Map, just drawing links as network connections rather than causal connections You can do that manually, or at scale with AI.

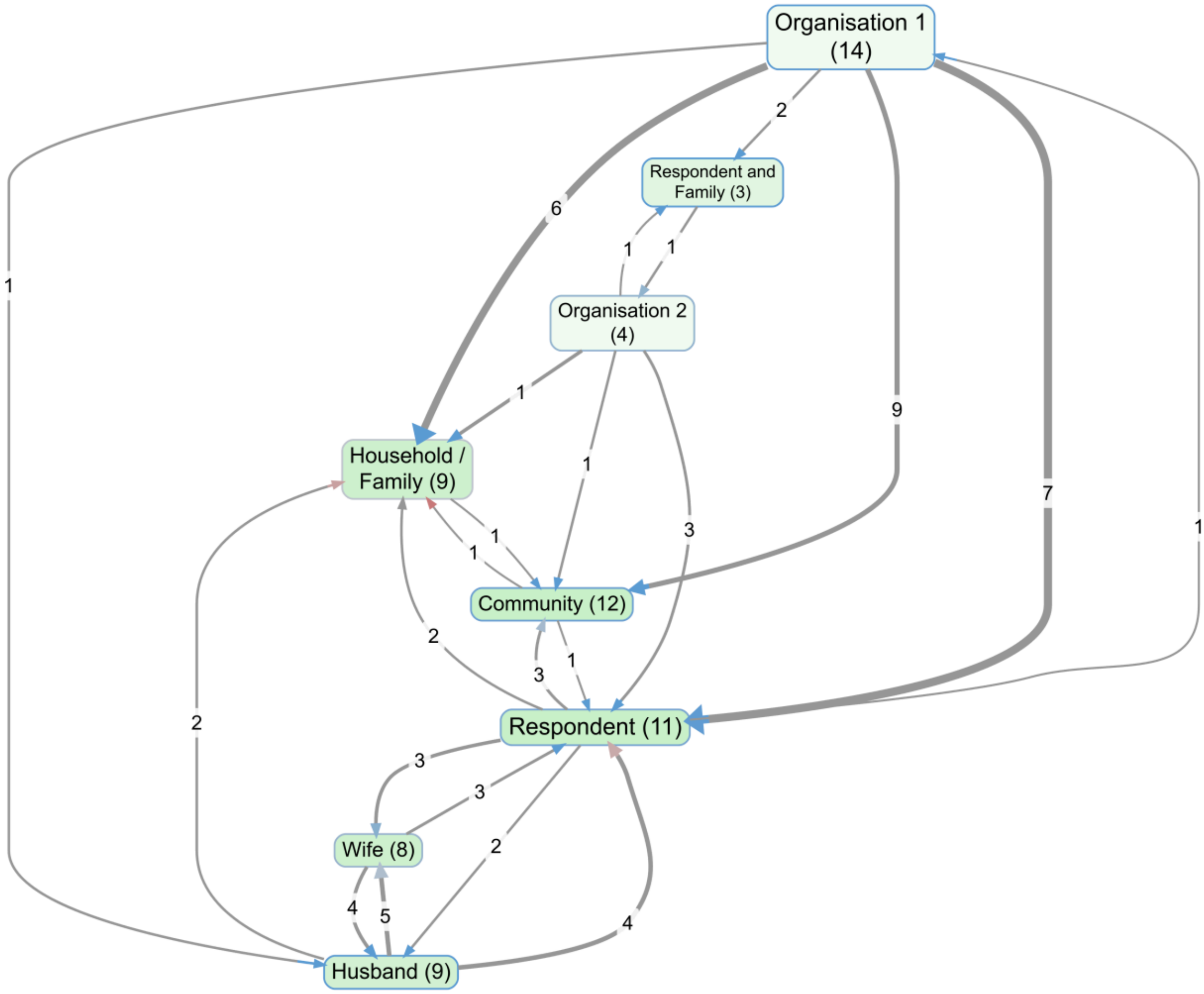
For comparison: A causal map

Here is a top-level view of a familiar anonymized International Development project, which we auto-coded as a **causal** map. If you want to play with it in Causal Map (it's free), this is the bookmark #1190.


Received training (Organisation 1) (12)
Improved/new farming techniques (12)
Increased knowledge (7)
Farm production (10)
Health behaviour (12)
Increased/maintained income (11)
Diet improved (11)
Food consumption quantity (7)
Improved health (10)
10
7
8
5
4
2
2
1
1
9
1
2
2
3
2
1
4
1
10
2
2
7
2
1
10
2
2
4
9
1

The social map

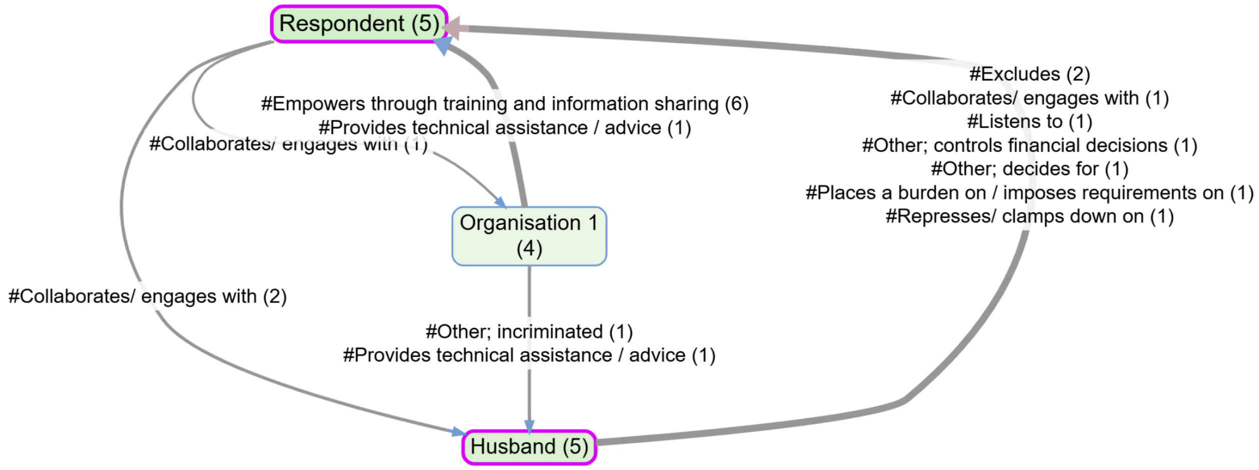
Here's the same set of interviews but coded as a kind of social network: *actor-focused*.



The coding instruction to the AI was very similar, just tweaking the instruction to focus relationship links rather causal links. Coding the 18 interviews took a few minutes. The big **difference is that in this case the links encode many different kinds of relationships**. Notice also that we've also automatically coded the sentiment of the associated link, and here the **average sentiment is shown as a colour** between red (negative) and blue (positive). Bookmark: **#1191.**

In Causal Map you can click on the individual links to see the quotes and explore the relationships.

We can show the different kinds of relationships on the links like this. This is part of a map filtered only to show female respondents. #1066.


Respondent (5)
#Empowers through training and information sharing (6)
#Provides technical assistance / advice (1)
#Collaborates/ engages with (1)
Organisation 1 (4)
#Excludes (2)
#Collaborates/ engages with (1)
#Listens to (1)
#Other; controls financial decisions (1)
#Other; decides for (1)
#Places a burden on / imposes requirements on (1)
#Represses/ clamps down on (1)
#Collaborates/ engages with (2)
#Other; incriminated (1)
#Provides technical assistance / advice (1)
Husband (5)

Pretty crass: the women say they collaborate/engage with their husbands but the arrow coming back to the respondents from their husbands is mostly negative.

Coding rule

We used this instruction for SNA coding of our standard `example-original` project.

```
You will receive texts from an international development project.

Try to understand the overall social network in these texts: who is related to whom
and how are they interrelated? then give me a standard table with columns, Sender,
Receiver, Relationship, Sentiment, Time, and Quote.



## Sender/receiver

Identify places in the text where it says that one (type of) social actor has a
relationship to another, and identify how these kinds of claims fit together into
chains or networks.

In the table you give me, each row is one relationship mentioned in the text. Find
common elements which appear as Sender and/or Receiver multiple times, forming chains
and other patterns that tell bigger stories, not just isolated pairs of links. In the
end, if one relationship really does not link up with anything else, it's ok to add
it to the table.

IT IS CRUCIAL THAT YOU UNDERSTAND THESE HAVE TO BE SOCIAL ACTORS (CITIZENS,
INDIVIDUALS, GOVERNMENTS, GOVERNMENT DEPARTMENTS, SPECIFIC GROUPS OF PEOPLE, FIRMS,
INSTITUTIONS ETC ETC ETC AND *NOT* NON-SOCIAL THINGS LIKE CLIMATE CHANGE, POSTERITY,
THE PAST, etc etc. We are only interested in RELATIONSHIPS AND TRANSACTIONS.

ALSO NOT A RELATIONSHIP BETWEEN ACTORS (and therefore not to be coded): larger than,
better than ... because these are abstract comparisons, not actual relationships or
transactions.



Sending and receiving are not meant physically. For example "the children were
suspicious of the policewoman" could be coded as Sender=children, Receiver=Police
Officer, Relationship=Other; suspicion.

All the links have a direction. If you find mutual relationships for example "the
children and the policewoman collaborated with another" then code two links, one in
each direction.

Do not include a link just because something happened and it vaguely involved two
actors. Only include it if this is primarily a relationship or transfer from one
actor to another.



### Relationship

```

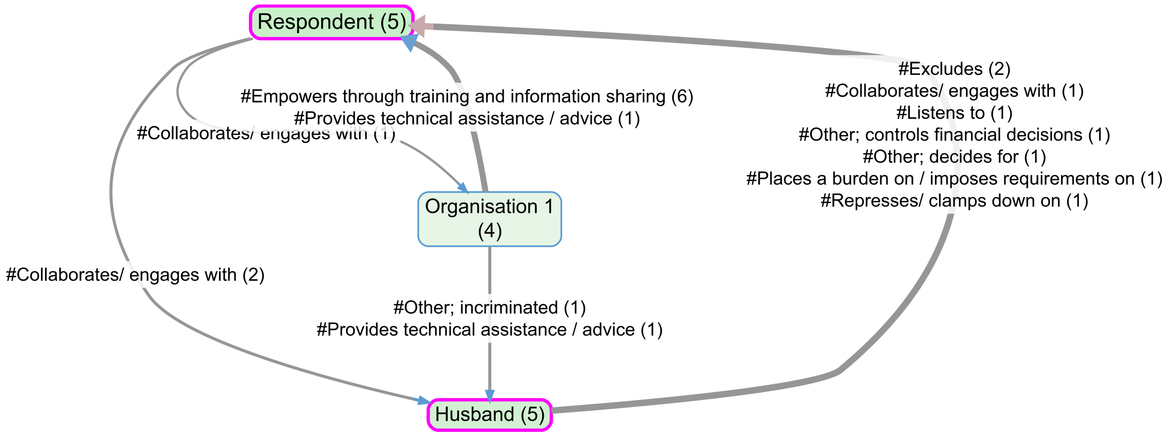
```
To characterise the relationship, you must choose one of these tags from the list.
Only use "Other" if the relationship really does not fit the main list. Also use this
as a final check that your link really is a relationship of some kind between two or
more (kinds of) actors. If not, skip this link.

#### List of allowed relationship tags.



"Collaborates/ engages with"
"Advocates to"
"Asks for help from"
"Enables implementation for"
"Listens to"
"Builds trust with"
"Empowers through training and information sharing"
"Provides technical assistance / advice"
"Provides funding and strategic support"
"Provides reporting support to"
"Represses/ clamps down on"
"Provides aid to e.g. food"
"Causes social friction/problems among"
"Maintains colonial pressure on"
"Exploits"
"Places a burden on / imposes requirements on"
"Excludes"
"Fails to collaborate with"
"Divides and misinforms"
"Other"



Do NOT invent your own tags. Choose from this list. Except that with the "Other" tag,
you can add details after a semicolon e.g. "Other; suspicious of". You MUST use
relational phrases like "Other; afraid of" not "Other; afraid". Do NOT use commas
ANYWHERE in your tags.



### Sender/receiver labels

Invent a label for sender and for receiver.



#### **Quote**

- The quote **must be a verbatim excerpt** from the text.
- **Do NOT modify or translate it.**

- **Do NOT break sentences apart or reassemble them differently.**

The quote should be long enough to be understood independently, usually 2-3 full
sentences with surrounding context. Do not provide a fragment only, provide enough
text that we can check if the entire causal claim with cause and effect are really
evidenced by this quote.
Do NOT use ellipses ("...")—only full, uncut text.



### Sentiment
1 = Positive 0 = Neutral or unclear, -1 = Negative .
Consider the context and the text's perspective when determining sentiment. Use 0 for
texts that have no clear positive/negative connotation, or when sentiment is
ambiguous. You can only add -1, 0, or 1 to the sentiment column, do not use words.

### Time
In this column, just put the time when the relationship happened / was prevalent

- Before and not during project
- During project
- Hypothetical / future





# Finishing

Now, for a technical reason, go back and change the Sender column to "Cause" and the
Receiver column to "Effect" and the Relationship column to "Tags". WE ARE STILL NOT
TALKING ABOUT ACTUAL CAUSATION AT ALL, FOR A TECHINCAL REASON WE HAVE TO SAY CAUSE
AND EFFECT INSTEAD OF SENDER AND RECEIVER.

So, your final column names are Cause, Effect, Tags, Sentiment, Time and Quote.

IT IS ESSENTIAL THAT YOU DON'T FORGET THE "Tags" column (which is renamed from
"Relationship") where you characterise the relationship between the actors. The links
are worth nothing to me if you miss off this "Tags" column.
```

Soft-recoding actors

…

Example filter

Respondent (female), Husband and Organisation 1: SNA.


Respondent (5)
#Empowers through training and information sharing (6)
#Provides technical assistance / advice (1)
#Collaborates/ engages with (1)
#Excludes (2)
#Collaborates/ engages with (1)
#Listens to (1)
#Other; controls financial decisions (1)
#Other; decides for (1)
#Places a burden on / imposes requirements on (1)
#Represses/ clamps down on (1)
Organisation 1 (4)
#Collaborates/ engages with (2)
#Other; incriminated (1)
#Provides technical assistance / advice (1)
Husband (5)

Filename: example-original-sna2. Citation coverage 8% of all sources: 20 citations shown out of 264. Factors — size: citation count; numbers: source count; colour: source count: in; border: avg incoming sentiment (blue=positive, grey=neutral, red=negative). Links — width: citation count; labels: unique tags (tally); arrowheads: effect sentiment (blue=positive, grey=neutral, red=negative). Filters applied: Sources included: All sources. Labels: Respondent, Husband; Source Groups: custom#Sex of the respondent=1; Factor freq: top 3 by source count. Bookmark #1066 2026-02-12 10:25_

Transformation and interpretation rules

Transformation rule

- **Input:** text describing relationships/transactions between social actors.
- **Transformation:** code each directed relationship as one row with role labels (sender/receiver), relationship tag, sentiment, time, and verbatim quote; then rename columns to app-compatible `Cause`, `Effect`, and `Tags`.
- **Output:** a links table in Causal Map format representing social-actor relationships.

Interpretation rule

- `Cause -> Effect` here denotes actor-to-actor relational direction, not abstract causal mechanism.
- `Tags` classify relationship type and should be read as relation semantics, not effect size.

Reporting global network statistics

4 Nov 2025

We do not actually provide these map-level statistics yet, e.g. "how connected overall is this whole map"?

Transformation and interpretation rules

Transformation rule

- **Input:** a full map/network.
- **Transformation:** global network-statistics computation is not currently implemented in the app.
- **Output:** no global network-statistics table/output yet.

Interpretation rule

- This extension is currently a placeholder/roadmap note.
- Users should interpret available factors/links metrics as partial substitutes until global stats are added.