🌻 Minimalist coding for causal mapping

Abstract

This paper explains our **Minimalist / Barefoot** approach to coding causal claims in text as simple directed links (“X influenced Y”), developed through large-scale practical coding and now implemented in the Causal Map app. We write it now because, although our previous work motivates causal mapping in evaluation (Powell et al., 2024), shows how QuIP-style “stories of change” elicit natively causal narrative evidence (Copestake et al., 2019), demonstrates ToC validation by comparing empirical maps with programme theory (Powell et al., 2023), and shows that genAI can extract links exhaustively with quotes as a low-level assistant (Powell & Cabral, 2025); (Powell et al., 2025), none of these papers is a standalone, reader-facing account of **the coding stance itself**.

Intended audience: evaluators / applied qualitative researchers who want a teachable causal coding protocol, and AI/NLP readers who want a simple, auditable target representation of causal content in text.

The contribution here is therefore to make the minimalist coding commitments explicit, teachable, and citable: what exactly counts as a coded causal claim, what we deliberately do *not* encode (effect size, causal typing, polarity-by-default), how evidence/provenance is kept auditable via quotes, and the limits of the approach (e.g. blockers/enablers and conjunctions). A companion formalisation paper A formalisation of causal mapping makes key parts of the method more precise (data structures, constraints, and conservative inference rules), but the focus here is the narrative rationale and practical coding guidance.

Unique contribution (what this paper adds):

- A definition of the **minimalist/barefoot** coding stance (“X influenced Y, with provenance”) and what it deliberately does *not* encode.
- An account of why this stance is useful at scale (including AI-assisted extraction) and where it fails (enablers/blockers; conjunctions/packages).
- Examples of additional extensions / transforms, called "filters" in the Causal Map app. Both the initial definition of a links table and each additional filter is described in terms of syntax and of semantics (interpretation rules).

As before, we treat causal mapping as three tasks

```
- Task 1, data gathering, not covered here;
- Task 2, coding: creating **evidence-with-provenance** (a links table)
- Task 3, analysis: a sequence of transforms of the original links table: the final interpretatio
```

Project context (how this paper fits)

This paper is part of a small set of new working papers

- Companion formal spec: A formalisation of causal mapping
- QDA positioning: Causal mapping as causal QDA
- Practical transforms/diagnostics: Magnetisation; A simple measure of the goodness of fit of a causal theory to a text corpus; Combining opposites, sentiment and despite-claims; other transforms to come later
- A worked “AI clerk vs human architect” example: Conversational AI -- Analysing Central Bank speeches

Introduction

I'll start by describing the "minimalist" approach to coding causal statements used for QuIP and developed originally by James, Fiona and colleagues at BathSDR and developed and further formalised at Causal Map Ltd in collaboration with BathSDR. This formalisation lives inside the Causal Map app. Then we will show how we can extend this approach with useful transformations. Finally I will try to answer the question of whether it can help us deal with more complicated constructions like enabling and blocking and whether this could help us with mid-range theory. As an appendix I'll add a more detailed overview of minimalist causal coding.

The minimalist approach is notable because it is based in our **joint experience of coding thousands and thousands of stakeholder interviews and other data such as project reports**, mostly from international development and related sectors, as well as coding hundreds of thousands of pages with AI-assisted coding. These have nearly always involved **multiple sources talking about at least partially overlapping subject matter**. So this coding produces individual causal maps for each source, which can then be combined in various ways -- rather than constructing single-source maps of expert thinking (Axelrod, 1976) or the collective construction of a consensus map (Barbrook-Johnson & Penn, 2022).

This paper sits alongside (and builds on) four related contributions. First, our evaluator-facing account argues that causal mapping is best treated primarily as a way to assemble and organise **evidence-with-provenance**, keeping the subsequent evaluative judgement about “what is really happening” distinct (Powell et al., 2024). We adopt that stance here: a coded link is first and foremost “there is evidence that a source claims X influenced Y”, not a system model with weights or effect sizes. Second, QuIP-style evaluation practice shows how “stories of change” can be elicited in a goal-free / blindfolded style to reduce confirmation bias, yielding narrative data that is natively causal (change plus reasons) and therefore well-suited to parsimonious link coding (Copestake et al., 2019). Third, our ToC comparison case study shows how empirical causal maps can be used as a disciplined way to check a programme’s Theory of Change against beneficiaries’ narratives, and to support evidence-based adjustment of “middle-level theory” rather than just project-level reporting (Powell et al., 2023). Fourth, our AI-assisted causal mapping work shows that this minimalist stance is also a practical sweet spot for automation: we can use genAI as a low-level assistant -- because the minimalist coding task is so relatively easy -- while keeping human judgement focused on the few high-leverage decisions (prompt design, verification, and synthesis choices) (Powell &

Cabral, 2025); (Powell et al., 2025). The present paper extracts and clarifies the core “minimalist” coding commitments that make that workflow workable and checkable.

Our experience has been that the vast majority of causal claims in these kinds of texts are easily and satisfactorily coded in the simplest possible form "X causally influenced Y". Explicit invocation of concepts like enabling/blocking, or necessary and/or sufficient conditions, or linear or even non-linear functions, or packages of causes, or even the strength of a link, are relatively rare. The causes and effects are not conceived of as variables, the causal link is undifferentiated, without even polarity, and if any counterfactual is implied it remains very unclear.

This approach is what we call **"Minimalist" or "Barefoot" Coding**.

Why minimalist coding?

Using more sophisticated, non-minimalist coding such as DAGs or fuzzy cognitive maps or whatever allows one to code linear or even non-linear causal influences of single or even multiple causes on their effects. One can do the "coding" by simply writing down (using appropriate special syntax) the connections, because one is an expert, and/or one can verify such statements statistically on the basis of observational data. Thus armed, one can make predictions or have sophisticated arguments about counterfactuals. But using minimalist coding we cannot do that, because our claims are formally weaker and therefore our inference rules are weaker. What we *can* do is still really interesting. We can ask and answer many different kinds of useful questions like:

- what are the main influences on (or effects of) a particular factor, according to the sources?
- what are the upstream, indirect influences on (or effects of) a particular factor, bearing in mind The transitivity trap?
- how well is a given programme theory validated by the respondents' narratives? (We can do this basically by using embeddings to get measures of semantic similarity between labels and aggregate these as a goodness of fit of theory to data.)

That is all exciting and useful. It's a surprisingly simple way to make a lot of sense out of a lot of texts which is, with caveats, almost completely automatable.

Minimalist coding principles

The 90% rule

We have found that it is pretty easy to agree how to apply minimalist coding to say 90% of explicit causal claims in texts, without missing out essential causal information, whereas it is very difficult to find appropriate frameworks to cope with the remaining 10%.

Fewest assumptions

Minimalist coding is perhaps the most primitive possible form of causal coding which makes no assumptions about the ontological form of the causal factors involved (the "causes" and "effects") or about *how* causes influence effects. In particular we do not have to decide if cause and/or effect is perhaps Boolean or ordinal, or if perhaps multiple causes belong in some kind of package or if there is some kind of specific functional relationship between causes and effects.

An act of causal coding is simply adding a link to a database or list of links: a link consists of **one new or reused cause label and one new or reused effect label**, together with the highlighted quote and the ID of the source.

A statement (S) from source Steve:

> I drank a lot and so I got super happy
>
> can be trivially coded minimalist-style as
>
> I drank a lot --> I got super happy (Source ID: Steve; Quote: I drank a lot and so I got super happy)

That's it.

Causal maps

Crucially, we can then display the coded claims for individuals as a graphical causal map, and we can also display the entire map for all individuals and/or maps filtered in different ways to answer different questions. There is a handful of other applications (Ackermann et al., 1996) (Laukkanen, 2012) for causal mapping which also do this; but as far as we know, only Causal Map also allows direct QDA-style causal coding of texts.

Data structure

Although we have the option of creating additional tags associated with each link (where many approaches would for example code the polarity of a link) this is not central to our approach.

We don't use a separate native table for factor labels: they are simply derived on the fly from whatever labels happen to be used in the current table of links. This makes data processing simpler and also suggests an ontological stance: causal factors only exist in virtue of being part of causal claims.

We do however have an additional table for source metadata including the IDs of sources, which can be joined to the links table in order, for example, to be able to say "show me all the claims made by women".

Causal powers

When a source uses causal language, we treat it as a claim that one thing made a difference to another (rather than a mere temporal sequence) *in virtue of its causal power to do so*.

Causal influence, not determination

We believe that it's rare for people to make claims about causal determination: someone can say that the heavy drinking made them super happy and then also agree that the music had a lot to do with it too, without this feeling like a contradiction.

Not even polarity

We differ even from most other approaches which are explicitly called "causal mapping" in that we do not even "out of the box" record polarity of links (to do so would involve making assumptions about the nature of the "variables" at each end of the link as well the function from one to the other).

The Focus on Cognition

In the minimalist approach, **we are quite clear that what we are trying to code is the speaker's surface cognitions and causal thinking**, while the actual reality of the things themselves is simply bracketed off at this stage, either to be revisited later (because we are indeed interested in the facts beyond the claims) or not (because we are anyway interested in the cognitions).

Staying on the surface

At Causal Map, we rarely make any effort to get beneath the surface, to try to infer hidden or implicit meanings. This is particularly well-suited to coding at scale and/or with AI. Our colleagues at BathSDR do this a bit differently, spending more effort to read across an entire source to work out what the source *really* meant to say.

Closer to the cognitive truth

It's really easy to code statements like (S) using minimalist coding. The trouble on the other hand with trying to use more sophisticated frameworks is that they are nearly always *ontologically under-determined*. For example, even a simple approach like Causal Loop Diagramming is function-based and requires at least a monotonic relationship between the variables: something like, *the more* I drink, *the happier* I get (in addition to which we have to code the actual, factual claims: I did drink a lot, and: I did get super happy). But is that what the speaker meant? How do we know if the speaker has say a continuous or Boolean model of "drinking"? If Boolean, what is the opposite of drinking a lot? Drinking only a little? If continuous, how do we know what kind of function they use in their own internal model?

We think it is often over-specified (and often psychologically implausible) to treat ordinary-language causal claims as if they asserted an explicit functional relationship between well-defined variables. Trying to force that kind of structure on everything turns the “easy 90%” of coding into a harder and more arbitrary task. Of course, one can decide to use a particular non-minimalist representation for a particular modelling purpose; our claim is only that this is usually not a faithful representation of what most speakers actually said or implied. For example, if we code "I got really tired because I have Long Covid", we could perhaps treat both endpoints as Boolean variables, but what about "I got really tired because it was really hot", and "I got really tired because it was really cold"—how are we going to represent “temperature” as a single variable while preserving the speaker’s intended meaning? If what we want to do is model a system, we can pick a solution. But if we want to model *cognition as expressed in text*, many “variable semantics” choices are over-committed.

Unclear counterfactuals

More formal, non-minimalist coding has clear counterfactuals. These may often be Boolean or continuous (the volume depends on the position of the volume dial; it's a 10, so the volume is maximum, if it had been at 5, the volume would have been about half as loud, and so on). Minimalist coding arguably implies some kind of naked counterfactual, but it is not always clear exactly *what*.

General versus specific

Minimalist coding focuses primarily on **factual causal claims** which also warrant the inference that both X and Y actually happened / were the case.

Most causal claims in the kinds of texts we have dealt with (interviews and published or internal reports in international development and some other sectors) are factual, about the present or past. Sometimes we see general claims, and we often just code these willy-nilly. In any case, the distinction between general claims and claims about specific events that actually happen is often fractal (a general claim can seem specific in a broader context) and difficult to maintain completely when modelling ordinary language.

We don't code absences

Minimalist coding may be reasonably also called **Qualitative Causal Coding** or **Causal QDA coding**. It shares characteristics with some forms of coding within Qualitative Data Analysis (QDA), in particular demonstrating an asymmetry between presence and absence (a specific tag not being applied is not the same as the application of an "oppositely-poled" tag.

We do not code absences unless they are specified within the text (e.g. perhaps "because of the barking dog, the owner did not come out of the house".

While codes may be counted, the concept of a *proportion* of codes is challenging because the denominator is often unclear.

If families are talking about reasons for family disputes, and family F mentions social media use, and family G mentions homework, we do not usually interpret this as meaning that family F does *not* think that homework can also be a cause of family disputes. (This should derive formally from the interpretation of multiple causal claims).

The labels do all the work

At Causal Map Ltd, our canonical methodology initially involves in vivo coding, using the actual words in the text as factor labels. This initial process generates hundreds of overlapping factor labels. This part is really easy (and is easy to automate with AI). Obviously, hundreds (or hundreds of thousands) of overlapping factor labels are not very useful, so we need to somehow consolidate them. Arguably, minimalist coding makes the initial coding easy but it just defers some of the challenges to the recoding phase.

!!Something about tags

Coping with many labels

We can:

- Use human- or AI-powered clustering techniques to consolidate the codes according to some theory or iteratively according to some developing theory
- Use AI-powered clustering techniques to consolidate the codes according to automated, numerical encoding of their meanings
- "Hard-recode" the entire dataset using a newly agreed codebook (see above)
- "Soft-recode" the dataset on the fly using embeddings to recode raw labels into those codebook labels to which they are most similar

Evidence strength is not causal effect size

Counts (how many times a link is coded; how many sources mention it) measure *evidence volume/breadth in the corpus*, not causal magnitude in the world. This matters because the outputs can look like a quantitative system model even when they are not one.

In the app we routinely distinguish:

- **Citation count**: how many coded mentions support a link (volume).
- **Source count**: how many distinct sources mention it (breadth / rough consensus).

These are useful for prioritising what to look at, or for filtering (e.g. keeping only links with `min_source_count = 2`), but they do not tell you whether a causal influence is “stronger” in any effect-size sense.

The transitivity trap and “thread tracing”

It is usually invalid to infer a long causal chain by stitching together links from different sources. A conservative rule is: only treat an indirect pathway (A \rightarrow B \rightarrow C) as supported when the *same source* provides each step ("thread tracing"), unless contexts are explicitly aligned.

The canonical failure mode is: source 1 says `A -> B`, source 2 says `B -> C`, and we mistakenly conclude that anyone told a coherent story `A -> B -> C`. In practice we handle this by "thread tracing": for a given query we loop through sources one at a time, construct the valid paths *within each source*, and then combine only the edges that appear in valid within-source paths.

The filter pipeline as a mental model for analysis

Most analyses are built by applying a sequence of simple operations (for example: subset links to retain only specific sources; trace paths; rewrite labels; bundle duplicates; then show a map/table). Thinking in terms of a pipeline helps make the meaning of an output explicit: it is always "the result of these filters, in this order".

Concretely, you can think of an analysis as "a links table passed through transforms", where `|>` is a "pipe" symbol which just passes a result from the left to a new transform on the right.

e.g.:

`Links |> filter_sources(...) |> trace_paths(...) |> transform_labels(zoom) |> combine_opposites |> bundle_links |> map_view`

This is also why "the same dataset" can yield very different-looking maps without any contradiction: you are simply looking at different derived views defined by different pipelines.

Positioning within qualitative research (and likely critiques)

This paper is about a **coding stance** and a corresponding **intermediate representation** (a links table with provenance), not a claim that "coding = analysis". In standard QDA terms, minimalist causal coding is best understood as a disciplined way to build an auditable evidence base that can later be interpreted, queried, and written up (Miles, Huberman, & Saldaña, 2014; Saldaña, 2021). It is, of course, not the only kind of way to do QDA.

To reduce avoidable points of attack from mainstream qualitative social science readers, we make four explicit commitments up front:

1. **We code claims, not causal truth.** A coded link is evidence that a source *claimed* an influence relation. It is not (by itself) a causal inference claim about the world. This keeps the method compatible with both realist and constructivist sensibilities (Lincoln & Guba, 1985; Charmaz, 2014).
2. **We do not treat "counts" as effect sizes.** Counting supports prioritisation and transparency, but it is not a substitute for interpretation; frequency/breadth in a corpus is not magnitude in the world. (This paper makes that distinction explicit below.)
3. **We trade interpretive depth for auditability and scale, on purpose.** We stay close to surface causal language and preserve provenance (quotes + source ids) so that readers can check what is being

claimed. This is a pragmatic stance when working with many sources and/or AI assistance; it does not deny that richer interpretive work can be valuable.
4. **Minimalist coding is not a full QDA workflow.** It is one layer that can sit alongside thematic analysis or qualitative content analysis when those are needed (Braun & Clarke, 2006; Mayring, 2000).

How a qualitative-methods reviewer might criticise this (and our response)

Below are common criticisms (often reasonable) and the corresponding guardrails/defences built into the minimalist stance.

Critique 1: "This is reductionist / strips context / decontextualises quotes"

Response: We keep provenance as first-class data (source id + quote) and treat every map/table view as a derived view of that evidence. Context is handled explicitly by joining source metadata (e.g. subgroup, time) and filtering the links table; it is not "assumed away". When deeper within-case interpretation matters, the same links table can be re-read within full-text context, and the write-up remains responsible for integrating that context (Miles, Huberman, & Saldaña, 2014).

Critique 2: "You are smuggling in a positivist epistemology (or a realist metaphysics)"

Response: The method's core object is **reported causal thinking** (surface causal claims), not an ontic causal model. We therefore keep the epistemic claim modest: "this is what sources said, with quotes", and we separate that from subsequent judgement about what is happening. That separation is consistent with standard qualitative criteria emphasising transparency, audit trails, and reflexive interpretation (Lincoln & Guba, 1985; Charmaz, 2014).

Critique 3: "Coding is not analysis; links don't 'explain' anything"

Response: Agreed. Minimalist causal coding produces a structured evidence base that supports multiple analyses (filters, comparisons, pathway queries) and supports more disciplined writing, but it does not replace interpretation. In practice, it can reduce time spent on low-level organisation of text so that more time can go into interpretation and argument (Saldaña, 2021; Gläser & Laudel, 2013).

Critique 4: "You are privileging causal language and missing other kinds of meaning"

Response: Yes: by design. Minimalist coding is for projects where the core questions are themselves causal (mechanisms, pathways, theories of change). When non-causal meaning is central (identity work, norms, metaphors), other QDA approaches could lead; minimalist coding can still be a useful *additional* layer rather than the whole analysis (Braun & Clarke, 2006).

Critique 5: "Automatable coding invites false precision and overclaiming"

Response: We explicitly constrain what automation is allowed to do: extract candidate links with quotes; keep everything auditable; and avoid treating the resulting network as a system model with polarity/weights “by default”. The transparency of provenance is the main defence against black-box “AI did the analysis” claims.

Extensions

An extension in general consists of syntax rules for how to construct and carry out a transformation, and semantic rules for what this transformation means.

There are many extensions one can add on top of minimalist coding. Small extensions can just add convenient functionality; other extensions can upgrade the system to other more fully-fledged kinds of coding like FCM.

This paper will only cover **hierarchical coding**, because it is simple, widely useful in practice, and it directly supports a transparent family of “zoom” transforms.

Extensions we do use

Hierarchical coding and “zooming” (FIL-ZOOM)

Motivation

In any qualitative coding process there is a tension between **detail** (keeping what respondents actually said) and **summarisation** (making patterns visible). Hierarchical factor labels give us a simple way to keep both:

- We can keep a detailed factor such as `Healthy behaviour; hand washing`.
- We can also treat it as an instance of a higher-level causal factor `Healthy behaviour` when we want a higher-level view.

This is particularly useful in causal mapping because it lets us simplify a complex map *without changing the underlying link evidence*: we are simply rewriting labels into more general ones.

Coding conventions

We use the separator `;` to create nested factor labels, using a template like:

```
General concept; specific concept
```

For example:

```
New intervention; midwife training -> Healthy behaviour; hand washing
```

This is an example of what we called earlier "letting the labels do all the work". As we do not in any case have a native factors table where we might record actual relationships between say lower-level and higher-level factors, we can do the same thing implicitly simply with the text of the label.

In AI-assisted coding, a practical way to encourage hierarchical labels is to explicitly instruct the coder (human or AI) to label each factor using the template **“general concept; specific concept”**, and to always provide a verbatim quote for each coded claim so every link remains checkable.

For example, an AI might produce a single coded link such as:

- `Improved agricultural practices; diversified crops -> Increased icome generation; more sales`
- Quote: “with a lot of good product, we are now able to sell more.”

You can read the separator as:

- “A, and in particular B” (forward), or
- “B, which is an example of / part of A” (reverse).

This can be extended to multiple levels:

`Improved hospital skills training; new midwife training; hand washing instructions improved`

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.

Zooming and visualisation

The basic idea of a zoom view is: **rewrite nested labels to a chosen level of abstraction**, then draw the map using the rewritten labels.

At “zoom level 1” we simply truncate at the first `;`:

- `Healthy behaviour; hand washing` becomes `Healthy behaviour`.

So if we have (at the detailed level):

`New intervention; midwife training -> Healthy behaviour; hand washing`

then we can show higher-level summary links such as:

- `New intervention -> Healthy behaviour; hand washing`
- `New intervention; midwife training -> Healthy behaviour`
- and (at the coarsest level) `New intervention -> Healthy behaviour`

Intuitively: by writing `A; B` you are explicitly licensing the interpretation “for high-level summary purposes, treat this as A”.

A worked example (one extension, end-to-end)

Suppose at the detailed level we code:

- `New intervention; midwife training -> Healthy behaviour; hand washing`

Then a zoom transform that truncates labels at the first `;` (zoom level 1) yields a higher-level summary view:

- `New intervention -> Healthy behaviour`

Crucially, zooming is not "inference about the world"; it is a **label rewrite** that produces a simpler derived view of the same evidence.

You can also see this as a family of conservative "deductions" licensed by the `;` separator. From:

- `New intervention; midwife training -> Healthy behaviour; hand washing`

we are explicitly allowing ourselves (for summary purposes) to treat it also as evidence for:

- `New intervention -> Healthy behaviour; hand washing`
- `New intervention; midwife training -> Healthy behaviour`
- `New intervention -> Healthy behaviour`

Caveats (don't use the `;` separator mechanically)

The `;` separator encodes a commitment: that the more specific factor can be "rolled up" into the more general one without essentially distorting the causal story. So:

- A factor cannot belong to **two different** hierarchies at once (because there is no single parent to roll up into).
- Higher-level factors should usually be **semi-quantitative causal factors** in their own right (not mere themes). If "Health" is just a topic tag, use a tag/hashtag convention rather than `Health; X`.
- Try not to mix desirability within one hierarchy (e.g. avoid `Stakeholder capacity; lack of skills`, because zooming out would treat it as evidence for "Stakeholder capacity"). Use opposites coding (`~...`) for this kind of case.

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

Do not use `;` merely to group factors into a non-causal topic theme. For example, these are both "health-related" but are not naturally special cases of a single semi-quantitative causal factor:

- `Vaccinations law is passed`
- `Mortality rate`

In that kind of case, prefer a light tagging convention (e.g. `#health` or `(health)`) rather than forcing a hierarchy like `Health; vaccinations law is passed`.

Other analysis filters (stubs; treated elsewhere)

These are best understood as steps in an explicit pipeline (as in the formalisation paper) and we will not develop them here.

Context filters (FIL-CTX)

Restrict the evidence base by selecting sources (e.g. only women; only a time period; only a subgroup), then carry out the same downstream analysis on the restricted links table.

Frequency / evidence-threshold filters (FIL-FREQUENCY)

Retain only links meeting a threshold of evidence strength (e.g. `min_source_count=2`). This is about *evidence volume/breadth in the corpus*, not effect size.

Topological / path-tracing filters (FIL-TOPO)

Retain only links that sit on paths connecting chosen factors (e.g. “mechanisms linking Training to Yield”), typically with thread tracing constraints.

Bundling (FIL-BUNDLE)

Compute evidence-strength columns for each `Cause -> Effect` pair (e.g. `Citation_Count`, `Source_Count`) and use those in map/table views.

Opposites coding (FIL-OPP) (stub)

Rather than encoding signed/weighted edges, we can sometimes get most of the benefit by treating explicit opposites in labels (e.g. `~Employment` vs `Employment`) as a label-level device with simple inference and visualisation rules.

AI extensions (optional; stubs)

Soft recoding (magnets) (FIL-SOFT)

Many raw in-vivo labels can be aggregated by mapping them to a smaller vocabulary of “magnets” using semantic similarity. This changes labels (a recoding step), not the underlying minimalist meaning of a single coded link.

Auto recoding (clustering) (FIL-AUTO)

Alternatively, labels can be clustered into emergent groups and then rewritten accordingly, again as a recoding/aggregation step rather than a new causal logic.

How other schools of causal mapping extend minimalist links

The “minimalist” stance in this paper is deliberately radical: we code *undifferentiated* causal influence claims with provenance, and then do most interpretation work via filters and views. Many other causal mapping traditions extend the representation in the opposite direction: they add more **semantic structure to factors and links**, especially **polarity** (and sometimes strength/weights), often with an implicit assumption that factors behave like **variables** and that a link expresses some kind of monotonic relationship. For a broader overview of these traditions and their differences, see (Powell et al., 2024).

What these extensions usually add

- **Polarity (+/−)**: a link can mean "more of X leads to more of Y" (positive) or "more of X leads to less of Y" (negative).
- **Strength / weights**: a link can be assigned a magnitude (sometimes elicited, sometimes assigned by analysts), which invites quantitative reasoning and simulation-like interpretations.
- **Variable semantics and counterfactual clarity**: polarity and weights usually presuppose that the endpoints are comparable as variables (or at least as ordered states), so that "more/less" (or "increase/decrease") is meaningful.

These extensions can be extremely useful in settings where respondents are explicitly reasoning in those terms, or where the purpose is modelling/simulation. But they also raise the bar: if we write down a signed or weighted link, we are committing not just to "X influenced Y", but to a stronger claim about *how* X and Y vary and how changes propagate.

Axelrod-style cognitive mapping (signed causal beliefs)

In the Axelrod tradition, cognitive maps are often treated as representations of beliefs about causal influence among concepts, and they are frequently coded with some notion of **positive/negative influence** in addition to direction (Axelrod, 1976); (Axelrod, 1976). This makes it easier to talk about reinforcing/balancing structure and about the direction of change, but it also moves the representation closer to variables-with-values (even if the original text/elicitation was not precise about scales or functional form).

Eden & Ackermann cause mapping / problem structuring

The Eden/Ackermann "cause maps" tradition (including its software lineage) emphasises causal mapping as a practical tool for structuring messy problems and supporting decision making, often built interactively with respondents and iterated in workshops (Eden et al., 1992); (Ackermann et al., 2004); (Ackermann et al., 1996). Polarity and more explicit causal typing are more natural here because the map is typically negotiated in context (so the group can decide what is meant by "increase/decrease", and can revise labels until the signed links make sense).

Comparative causal mapping (standardisation and cross-map comparison)

Comparative approaches focus on how to elicit, standardise, and compare large numbers of maps across people, groups, or time. This tends to bring in more explicit conventions for coding and comparison, and often assumes a more "variable-like" interpretation of factors so that maps can be aligned and analysed at scale (Laukkanen, 1994); (Laukkanen, 2012); (Markiczy & Goldberg, 1995); (Hodgkinson et al., 2004).

What it would take to extend our logic to signed links (and why we treat it separately)

We *can* extend our links-table logic to incorporate polarity, but doing so is not just "adding a column"; it changes the semantics.

At minimum, we would need:

- A `Polarity` field per coded link (e.g. `+`, `-`, `unknown`) that is grounded in the source text or elicitation.
- A rule for how polarity is inferred when it is implicit, ambiguous, or mixed (which is common in narrative material).
- A clear semantics for what `+` and `-` *mean* (typically: a monotonic relationship between variables), including what counts as "more/less" for a factor label.
- Aggregation rules for how to deal with disagreement and contradiction across sources, and for how to visualise those disagreements without creating spurious precision.

Because our target corpora typically do *not* make those commitments explicit (and because we are focused on modelling evidence-with-provenance rather than system dynamics), we do not treat signed links as part of the core method here. Where polarity matters in practice, we prefer to handle it with **label-level devices** (e.g. `~Employment` vs `Employment`) and with explicit, auditable transformations (e.g. `combine_opposites`), and we treat "signed/weighted link semantics" as a separate family of approaches to be discussed elsewhere.

What we cannot do

None of this really answers all the questions raised above about problematic cases such as what to do with "I got really tired because it was really hot", and "I got really tired because it was really cold" or any other case where we have different factor codes which have shared information. At first blush, this isn't a problem, we can simply code "it was really hot" and "it was really cold" separately, but how to parse the contents to reflect the fact that these two are related? Or, how to parse the contents of "Improved health behaviour (hand washing)" and "Improved health behaviour (using a mosquito net)" to reflect the fact that they are somehow neighbours? We do have some tricks for this, but that would take us beyond the present discussion.

Causation about causation: enablers and blockers as second-order causal claims

One natural way to try to go beyond truly minimalist coding is to say that some claims are not about simple factors causing other simple factors, but about one factor influencing (enabling or blocking) a *causal connection* between two other factors.

Consider an enabler-style statement:

> I went on holiday to Spain expecting to enjoy it, and indeed I did particularly enjoy it because I remembered to take my phrase book.

In strictly minimalist coding, we might record two undifferentiated links:

- Going on holiday to Spain --> Enjoying the holiday

- Remembering to take my phrase book --> Enjoying the holiday

But arguably the second claim is not really “the phrase book caused enjoyment” in the same way as the first. Rather, it enabled the normal causal power of the holiday to produce enjoyment. A more explicit encoding would therefore be:

1. Going on holiday to Spain --> Enjoying the holiday
2. Remembering to take my phrase book --> (Going on holiday to Spain --> Enjoying the holiday)

Visually, (2) would be drawn as an arrow pointing to the middle of the arrow in (1). Semantically, it can be read as:

> The phrase book enabled the causal link from X to Y, in virtue of its causal power to do so.

Blockers are closely related but show the asymmetry more starkly:

> I went on holiday to Spain, but I forgot to take my phrase book so I didn't enjoy the holiday at all.

One can code:

- Forgetting to take my phrase book --> Not enjoying the holiday

However, our intuition that “going on holiday” should also be part of the story is hard to capture without moving to a second-order encoding. The blocker case is visually similar to the enabler case, but it seems to require adding the idea of a causal power to *stop* something: the blocker destroys or disables the causal power of X to bring about Y.

I do not claim we can do much with this at scale; but it is a clear way of stating what seems to be “missing” from purely first-order, undifferentiated links in these edge cases.

Causal packages / conjunctions

Another class of difficult cases is causal packages: claims in which some *combination* of factors is said to be needed for an effect.

For example, “you need an accelerant and a spark to set a fire” is not well represented by coding two separate links (“accelerant causes fire” and “spark causes fire”), because that misses the conjunctive structure. One can code the cause as a single phrase (e.g. “an accelerant and a spark”), but then the phrase is not parsable in a way that lets us relate it to other claims about accelerants or sparks on their own.

In principle one could introduce special syntax for conjunctions, but the moment we do so we are immediately pushed towards stronger, more model-like commitments (e.g. about truth tables, interaction effects, non-linear combination rules), and it is unclear how much such structures would recur with enough regularity in ordinary language corpora to justify that added complexity.

Blockers and enablers

Maybe we could ascend from formally weaker but numerically overwhelming minimalist-coded data to make other rich conclusions, in particular about enablers and blockers like the headphones and the rain. However, I don't think this is really possible. In minimalist coding, at the level of individual claims, you can code "The headphones enabled James to answer the question in the Zoom call" as

> The headphones --> James was able to answer the question in the Zoom call

... but we cannot easily get inside the *contents* of the effect. We might like to be able to code this as the effect of the headphones not on a simple causal factor but on *another causal connection*, namely between the question on the Zoom call and James' answer, but we do not have any way at the moment to do this. It might be possible to extend minimalist coding to cope with this, perhaps ending up with three factors (headphones, question, answer) and some new syntactic rules to code their relationship, and some corresponding new semantic rules to be able to deduce more things about these three factors, but I think this would be **missing the point**. I'm not sure what we could do with these kinds of subtle relationships at any scale. Let's guess that within a given corpus, five percent of causal claims are of this form: what are the chances of such claims then overlapping enough in content that we could then apply our new more specialised deduction rules in more than a handful of cases?

It might be the case that certain specific more sophisticated causal constructions become **part of ordinary language**. For example: "Her post mocking Farage went viral, so Farage was forced to respond". Here, the concept of *going viral* is perhaps a kind of shorthand for a quite sophisticated causal claim, yet it might be common enough for us to be able to usefully code it (and reason with it) using only unadulterated minimalist coding, without causally unpacking "her post went viral". So that's useful, and maybe it is even useful in building some kinds of mid-range theory, but without actually understanding or unpacking what "going viral" means.

So that's it, in a nutshell. Sorry to disappoint, James.

See also:

(Powell et al., 2024)

(Powell & Cabral, 2025)

(Britt et al., 2025)

(Powell et al., 2025)

(Remnant et al., 2025)

Selected references (APA 7; added for qualitative-methods positioning)

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Charmaz, K. (2014). *Constructing grounded theory* (2nd ed.). SAGE.

Gläser, J., & Laudel, G. (2013). Life with and without coding: Two methods for early-stage data analysis in qualitative research aiming at causal explanations. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research, 14*(2). https://www.qualitative-research.net/index.php/fqs/article/view/1886

Gläser, J., & Laudel, G. (2019). The discovery of causal mechanisms: Extractive qualitative content analysis as a tool for process tracing. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research, 20*(3). https://doi.org/10.17169/fqs-20.3.3386

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. SAGE.

Mayring, P. (2000). Qualitative content analysis. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research, 1*(2), Art. 20. https://www.qualitative-research.net/index.php/fqs/article/view/1089

Miles, M. B., Huberman, A. M., & Saldaña, J. (2014). *Qualitative data analysis: A methods sourcebook* (3rd ed.). SAGE.

Saldaña, J. (2021). *The coding manual for qualitative researchers* (4th ed.). SAGE.

[Note: we will add at least the formal codes like combine_opposites from the formalism paper]

References

Ackermann, Jones, Sweeney, & Eden (1996). *Decision Explorer: User Guide*. https://banxia.com/pdf/de/DEGuide.pdf.

Ackermann, Eden, & Cropper (2004). *Getting Started with Cognitive Mapping*.

Axelrod (1976). *Structure of Decision: The Cognitive Maps of Political Elites*. Princeton university press.

Axelrod (1976). *The Analysis of Cognitive Maps*. In *Structure of Decision : The Cognitive Maps of Political Elites*.

Barbrook-Johnson, & Penn (2022). *Participatory Systems Mapping*. In *Systems Mapping: How to Build and Use Causal Models of Systems*. https://doi.org/10.1007/978-3-031-01919-7_5.

Britt, Powell, & Cabral (2025). *Strengthening Outcome Harvesting with AI-assisted Causal Mapping*. https://5a867cea-2d96-4383-acf1-

7bc3d406cdeb.usrfiles.com/ugd/5a867c_ad000813c80747baa85c7bd5ffaf0442.pdf.

Copestake, Morsink, & Remnant (2019). *Attributing Development Impact: The Qualitative Impact Protocol Case Book*. March 21, Online.

Eden, Ackermann, & Cropper (1992). *The Analysis of Cause Maps*. https://onlinelibrary.wiley.com/doi/10.1111/j.1467-6486.1992.tb00667.x.

Hodgkinson, Maule, & Bown (2004). *Causal Cognitive Mapping in the Organizational Strategy Field: A Comparison of Alternative Elicitation Procedures*.

Laukkanen (1994). *Comparative Cause Mapping of Organizational Cognitions*.

Laukkanen (2012). *Comparative Causal Mapping and CMAP3 Software in Qualitative Studies*. https://doi.org/10.17169/fqs-13.2.1846.

Markiczy, & Goldberg (1995). *A Method for Eliciting and Comparing Causal Maps*. Sage Publications Sage CA: Thousand Oaks, CA.

Powell, Larquemin, Copestake, Remnant, & Avard (2023). *Does Our Theory Match Your Theory? Theories of Change and Causal Maps in Ghana*. In *Strategic Thinking, Design and the Theory of Change. A Framework for Designing Impactful and Transformational Social Interventions*.

Powell, Copestake, & Remnant (2024). *Causal Mapping for Evaluators*. https://doi.org/10.1177/13563890231196601.

Powell, Cabral, & Mishan (2025). *A Workflow for Collecting and Understanding Stories at Scale, Supported by Artificial Intelligence*. SAGE PublicationsSage UK: London, England. https://doi.org/10.1177/13563890251328640.

Powell, & Cabral (2025). *AI-assisted Causal Mapping: A Validation Study*. Routledge. https://www.tandfonline.com/doi/abs/10.1080/13645579.2025.2591157.

Remnant, Copestake, Powell, & Channon (2025). *Qualitative Causal Mapping in Evaluations*. In *Handbook of Health Services Evaluation: Theories, Methods and Innovative Practices*. https://doi.org/10.1007/978-3-031-87869-5_12.