Finding your MUSE: Mining Unexpected Solutions Engine

Anonymous ACL submission

Abstract

Innovators often exhibit *cognitive fixation* on existing solutions or nascent ideas, hindering the exploration of novel alternatives. This paper introduces a methodology for constructing Functional Concept Graphs (FCGs), interconnected representations of functional elements that support abstraction, problem reframing, and analogical inspiration. Our approach yields large-scale, high-quality FCGs with explicit abstraction relations, overcoming limitations of prior work. We further present MUSE, an algorithm leveraging FCGs to generate creative inspirations for a given problem. We demonstrate our method by computing an FCG on 500K patents, which we release for further research. A user study indicates that participants exposed to MUSE's inspirations generated more creative ideas, both in terms of absolute number (up to 19% increase over participants not given inspirations) and ratio (75%, compared to 49% for no inspirations).

1 Introduction

A well-documented challenge in design and problem-solving is *fixation*, wherein individuals become prematurely attached to a narrow set of familiar solutions or features, thereby impeding the generation of truly novel concepts (Jansson and Smith, 1991; Purcell and Gero, 1996). This cognitive inertia can significantly limit the effective exploration of the design space – the abstract space of all possible solutions to a given problem. Navigating this complex space to identify diverse and high-quality solutions requires systematic approaches that encourage cognitive flexibility and the consideration of a broad range of alternatives.

Existing methodologies for ideation often fall short in robustly guiding designers out of fixation traps or in structuring the vastness of the design space in a functionally meaningful way. While keyword-based search can retrieve superficially related concepts, it often fails to uncover *deeper functional analogies* that are crucial for creative leaps (Holyoak and Thagard, 1996). There is a growing need for representations that capture the core functional essence of ideas, enabling a more principled exploration of potential solutions.

We build upon an idea explored in recent work, that suggested decomposing ideas into their fundamental *purposes* (problems) and *mechanisms* (solutions) and organizing these into a structured representation called a *Functional Concept Graph* (FCG) (Hope et al., 2022). In FCGs nodes correspond to purposes and mechanisms of products; edges either link between purposes and mechanisms that achieve them, or between purposes and their *abstraction* (i.e., a more general problems).

We posit that such a representation can serve as a powerful cognitive scaffold for designers and problem-solvers. Specifically, by enabling navigation across interconnected functional elements, it facilitates the abstraction of problem statements and the discovery of *analogical inspirations* from domains that might seem unrelated at the surface

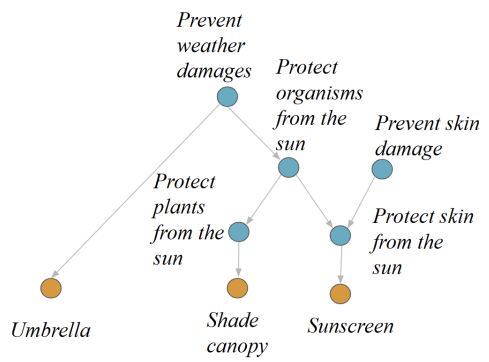

Figure 1: An example of a Functional Concept Graph. Node represent problem (spurposes, in blue) and solutions (mechanisms, in orange). Connections between problem nodes indicate abstraction. Connection between solution and problem nodes indicates the mechanism can solve the problem.

level (Gentner, 1983; Gick and Holyoak, 1980).

For example, see Figure 1. Suppose an inventor wishes to protect their plants from the sun. They are familiar with standard mechanisms, such as shade canopies. Through the graph, they might discover analogous mechanisms linked to the same abstract purpose. For example, they might reach the purpose node "protect skin from sun" through shared parent "protect organisms from the sun", which might inspire them to think of sunscreen for plants (which, surprisingly, has already been invented). Alternatively, they might explore even higher-level abstractions. This structured exploration can systematically help designers break free from initial conceptual ruts, reframe their understanding of the problem, and ultimately chart more innovative trajectories within the design space.

However, the implementation of Hope et al. (2022) was very limited. Most notably, their edges only capture simple co-occurrence patterns in the corpus; there is no guarantee of explicitly encoding abstraction. In addition, their annotation process relied heavily on crowdworkers, which resulted in a noisy graph, and graph-building did not scale.

In this work, we take advantage of the tremendous recent progress of LLMs and reimagine Functional Concept Graphs. Our contributions are:

- We propose a novel, scalable approach to constructing Functional Concept Graphs that results in richer, better-connected and less noisy graphs, whose edges explicitly encode abstraction relations.
- We compute an FCG on 500K patents, and release it for further research.
- We introduce MUSE, an algorithm that, given an FCG and a target problem, can produce inspirations for creatively solving the problem.
- We conduct a user study and show MUSE inspirations can enhance human creativity. Our analysis shows that using our inspirations, participants were able to come up with up to 19% more creative solutions, in comparison to participants that did not receive inspirations. More importantly, 75% of the solutions produced by participants exposed to MUSE's inspirations were deemed creative (compared to 49% for participants who did not see the inspirations).
- We release all the data and code used for creating the graph and analysis. [1]

[1] Code and data redacted for the anonymity period.

2 Functional Concept Graphs

Our goal is to automatically build a Functional Concept Graph (FCG) (Hope et al., 2022) and develop an algorithm to sample inspirations from it, given a target problem.

Building upon the foundations of functional modeling, an FCG provides a structured graphical representation. Nodes embody functional concepts, encompassing both the intended purposes (problems tackled) and the underlying mechanisms (solutions) that enable these purposes. A directed edge between a purpose node and a mechanism node indicates that the mechanism is useful for achieving the purpose; a directed edge between two purposes indicates that the first is an abstraction of the second (see Figure 1).

3 Data

We chose to test our idea on a patent corpus because it is vast, publicly available and contains a variety of real-life problems and solutions. In contrast, Hope et al. (2022) has used a much smaller dataset of crowdsourced innovations.

We use a dataset taken from Patentsview.org website (Toole et al., 2021), which contains patents registered in the U.S. since the 1940s. From each patent, we take its title and abstract text, which contains an informative description of the product. In addition, as part of our annotations, we used each patent **CPC** tag.

CPC (Cooperative Patent Classification) is a hierarchical patent classification system, developed by the European and US Patent Offices. The system assigns each patent with (potentially multiple) CPC tags. Each tag has an id and a short name.

Out of the full patent corpus extracted from the website, we included only patents that belong to 3 CPC top-level tags (out of 9 overall): Human Necessities (A), Operations and Transport (B), and Mechanical Engineering (F). We included those tags as they are likely to describe relatively simple everyday products, as opposed to sections that describe more specific professional domains such as Chemistry and Metallurgy (C). After the filtering, we were left with about 3M patents. Due to a budget limitations, we sampled 500K patents and used them as our corpus.

4 Constructing an FCG

Our pipeline consists of 3 main steps (see Figure 2): (1) extracting problems and solutions from a large

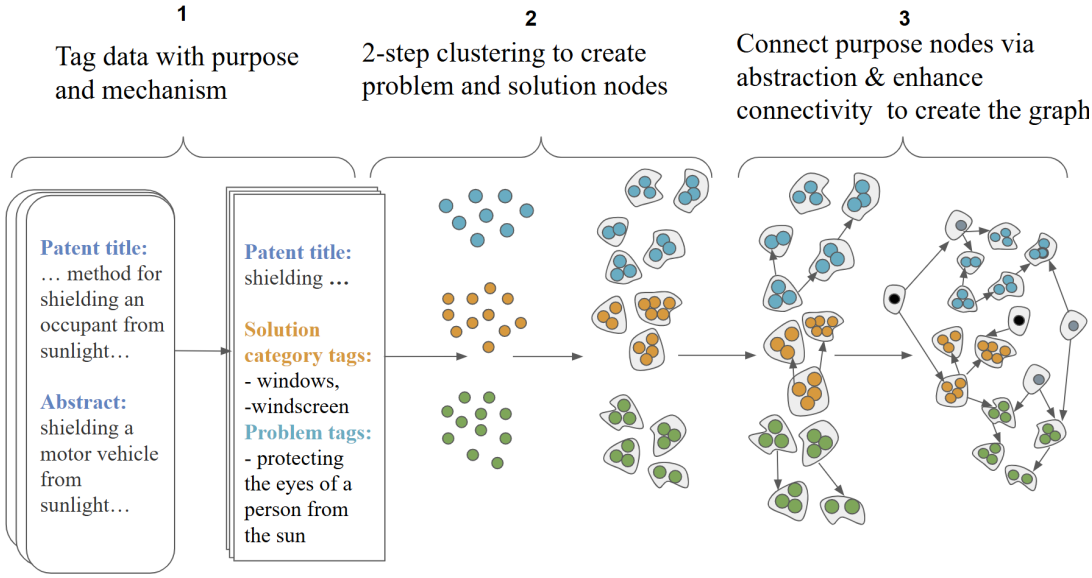

Figure 2: A visualization of our full pipeline. (1) We start by extracting purpose and mechanism tags from patent descriptions. (2) Then, we create the problem and solution nodes. To reduce the amount of computation, we first cluster the purpose tags to create loose clusters, then aggressively cluster each loose cluster to obtain the problem nodes. This clustering induces the solution nodes as well. (3) The final step is to create edges connecting the problem nodes by finding abstraction relations, and then enhacing connectivity through virtual nodes.

dataset of patents, (2) creating the nodes of the FCG and (3) adding in the edges.

4.1 Extracting problems and solutions

We start by annotating each patent with its *purpose* (problem, what is it used for) and *mechanism* (solution, how it works). We acquire multiple mechanism and purpose tags for each patent, as opposed to a single aggregated tag.

Importantly, patents are written in technical, legal language ("Legalese"). Surprisingly often, they are missing important commonsense information (for instance, a patent about airbags that never mentions cars or accidents, but rather focuses on the technical aspects of the invention). Thus, we take advantage of large-scale language models as well as patent metadata to annotate the dataset.

Getting mechanisms. To extract mechanism tags for each patent, we make use of the CPC tags, that offer a granular breakdown of the technical features of the invention.

We processed the full set of CPC categories (over 250,000 categories). We discard all CPC tags in the lowest level of hierarchy level, since they are too specific for our needs. We clean and preprocess the titles (see Appendix C.2), and end up with a set of 8500 CPC tags.

Many CPC tags describe mechanisms, but not all. Thus, we manually tagged 1500 CPC tags as either related to mechanism or not. We used this annotated dataset to fine-tune a simple RoBERTa-based binary classification model (Liu et al., 2019). We train the model for 500 epochs, using 1e-7 learning rate. The final model we used achieved an F1 score of 0.88.

Getting purposes. Driven by its few-shot capabilities, we use GPT3 (Brown et al., 2020) to generate *purpose* tags. We adopt an in-context learning approach, and use the method proposed in Reif et al. (2021) to construct a prompt with 3 patent descriptions and their purpose tag annotations, followed by the patent description to be tagged (see Figure 2 for example tags and Appendix B for prompt example). Although newer and more advanced models are available, we found that GPT3 (Babbage) provided the best performance-cost tradeoff for tagging 500K patents.

4.2 Creating problem and solution nodes

After obtaining the purpose and mechanism tags, we move on to creating the problem and solution nodes that serve as our basic building blocks for the graph.

Creating purpose nodes. Our goal is to create problem nodes that cluster together conceptually similar purpose tags. To achieve this goal, we choose to use agglomerative clustering (Ward Jr, 1963) over Sentence-BERT (Reimers and

Gurevych, 2019) embeddings of the purpose tags. Agglomerative clustering allows us to control the similarity threshold, and also does not require specifying the number of clusters in advance.

However, running agglomerative clustering over all purpose tags is costly. Thus, we perform clustering in two phases; we use K-means (MacQueen, 1967) to split the tags into loose clusters, and then run agglomerative clustering on each of them. This idea is to have agglomerative clustering with a strict similarity threshold operate on smaller, manageable sets of tags, making sure to keep semantically similar tags in the same set. We use the resulting clusters as the problem nodes in our graph.

Creating solution nodes. We wish to create clusters of solutions used to solve similar problems. These solutions are not necessarily semantically similar themselves. Therefore, we do not cluster the mechanism tags directly, but rather induce the solution clusters from the problem clusters we created. We cluster two mechanism tags together if there exist purpose tags extracted from their corresponding patents that were clustered together in the previous step.

4.3 Adding edges

The final step of creating the FCG is to connect the problem and solution nodes we obtained in section 4.2.

As described in section 2, edges in FCGs reflect either an abstraction relation between problem nodes or a problem-solution relation.

Problem to solution. We connect a problem node to a solution node if there is at least one patent that was assigned a problem tag in the problem node (i.e., in the corresponding cluster) and a solution tag in the solution node.

Problem to abstract problem. Intuitively, if a problem ("protecting an organism") is more abstract than another ("protecting plants"), the specific problem *entails* the more abstract one. Hence, we use a pretrained NLI model (Laurer et al., 2024) to identify entailment. As checking for entailment over all problem nodes is computationally expensive, we run the model over all pairs of problem nodes from the same loose cluster (Section 4.2). For each problem node, we select a representative purpose tag and add a prefix to turn it into a sentence (so it resembles the data the NLI model was trained on). We add an edge if the entailment score is above threshold t. See Appendix C.4 for details about entailment thresholds and prefixes.

We note that the graphs formed by this process might include cycles, due to mistake or inconsistencies of the NLI model. Moreover, these graphs might also include redundant edges (if the graph contains edges $(x_1, x_2), (x_2, x_3)$, edge (x_1, x_3) is redundant). Therefore, we adopt the method suggested in Sun et al. (2017) to remove cycles while maintaining the abstraction hierarchy. Then, we eliminate redundant edges in the graph by keeping only the longest paths between any connected nodes. The result of this process are K interim graphs $G_1, ..., G_K$ (one for each K-mean cluster).

Enhancing connectivity. Some abstract relations might not be captured by the NLI model. Thus, we create two types of *virtual* nodes to enhance the connectivity of problem nodes:

- **LLM-based connections.** We first seek abstraction relations that were not captured by the NLI model. For each loose-cluster graph G_i, we select a set of candidate nodes N_i that describe relatively general problems, based on their height in the graph. We use llama3.1-8b-Instruct (Grattafiori et al., 2024) to suggest abstractions that capture several of these nodes. We create a virtual LLM-node for each suggested abstraction, and connect it with the nodes responsible for its creation.
- **Verb-based connections.** To capture far analogies in the graph, we choose to abstract problem nodes by the *verb* appearing in them (e.g., protect). We collect lists of synonymous verbs from WordNet (Fellbaum, 2010), and create a virtual verb-node for each list of synonyms. Then, we connect each node in the graph to all verb-nodes containing verbs appearing in it.

The final step is to try to explicitly improve the connections between the interim graphs $G_1, ..., G_K$, to allow the user to find *far analogies*. We repeat the process of enhancing connectivity, but this time we focus on pairs of nodes coming from *different* sets of candidate nodes, $u \in N_i, v \in N_j, i \neq j$. See details in Appendix C.6.

5 MUSE: Sampling inspirations

Given a starting node problem n_p, we wish to sample inspirations from the graph. The graph provides an intuitive interpretation of paths: For example, one could move up and then down, reaching a sib-

ling node (connected by a shared parent abstraction). Although many types of paths might produce useful inspirations, we limit the paths from which we sample inspirations to ("up", "down") and ("up", "up", "down"). Sampling from other paths is left for future work.

Sampling nodes from different sources. In Section 4.3 we described 3 types of connections between purpose nodes: NLI-based, verb-based and LLM-based. We define *LLM-nodes* (*verb-nodes*) as nodes for which at least one of the edges along the path connecting then with n_p is LLM-based (verb-based). NLI-nodes are nodes for which all of the edges along the path are NLI-based.

Ensuring diversity. Consider all nodes reachable from the start node n_p by following paths ("up", "down") or ("up", "up", "down"). To ensure the sampled nodes are both relevant and diverse, we use MMR (Carbonell and Goldstein, 1998) to select up to 5 nodes from each path and source (up to 30 nodes total). We refer to this sampling process as MUSE – an algorithm aimed at helping users come up with novel ideas.

6 Evaluation

Now that we have built our graph, our goal is to use it to help users find creative and novel solutions to their problems. Specifically, we are interested in the following research questions:

RQ1: Can inspirations from the FCG enhance users' ability to come up with original solutions to problems? If so, what is the best way to communicate the inspirations to users?

RQ2: Which type of trajectories in the FCG produces more helpful inspirations?

6.1 Experiment design

To answer these questions, we conducted a user study. Participants were randomly assigned a problem. After reading the instructions, they were given 15 minutes to come up with as many creative ideas as they could. Each participant was randomly assigned one out of 4 conditions, corresponding to different ways to display the inspirations (see below). In 3 of the conditions participants received inspirations drawn from the FCG; in the last condition no inspirations were given. The subjects that received inspirations were instructed to identify the source of inspiration for each ideas (which could be their own idea or one of the inspirations provided). The full instructions for the experiment are provided in Appendix E.

The experiment was carried out on the Prolific platform (Prolific, 2014). Participants were paid £3.25. 61 native English-speakers took part in the experiment, split almost evenly between conditions (16-15-15-15) and problems (31-30). 53.3% identified as females, and 46.7% as males. 21.4% were in the 18-29 age group, while the percentages for the age groups 30-44, 45-59 and 60+ were 42.9, 28.6 and 7.1, respectively.

Problems. We chose 2 everyday problems: "Seal a leak" and "Cool a room". The problems were selected from a list of everyday household problems from ehow.com. We chose these problems since they are very familiar to the common person, have well-known existing solutions while still enabling creative solutions.

For each problem p, we encode the text describing it using Sentence-BERT, and then find the closest node to it in the graph n_p using Faiss-index (Douze et al., 2024).

Displaying the inspirations. Inspirations are problems sampled from the graph as described above. We define 4 conditions (ways to display them):

1. **Purpose**: Showing just the purposes.
2. **Purpose + mechanism**: Showing purposes + up to 3 solutions sampled for each purpose.
3. **Purpose + mechanism sentence**: Same as condition 2, but we use an LLM (Claude 3.7 Sonnet (Anthropic, 2025)) to turn inspirations into full sentences.
4. **Empty** No inspirations.

One example of a problem sampled as inspiration for the problem "Cool a room" is *"Creating a chamber seal mechanism"*. A solution sampled for this problem is *"Packing rings"*, and the generated sentence is "Packing rings expand under fluid pressure to create effective chamber seals". Additional examples are given in Appendix D.

7 Results

We are interested in the degree to which our approach helps users to come up with creative solutions to well-known problems. For that, we measured the quality of the solutions produced by the participants in the experiment. Expanding upon Reinig et al. (2007); Hope et al. (2017), we define creativity as a combination of **utility** and **novelty**. Thus, we first score each solution with a binary feasibility score representing whether this solution is feasible and solves the given problem. Feasible

	Purp	Purp+ mech	Purp+ mech sent	Empty
Feasible (#)	4.37	3.67	**4.8**	**4.8**
Feasible (%)	0.72	0.78	**0.83**	0.63
Creative (k=2) (#)	3.18	2.6	**4.06**	3.4
Creative (k=2) (%)	0.58	0.57	**0.75**	0.49
Novel(k=3) (#)	1.81	1.13	**1.86**	1.73
Novel(k=3) (%)	0.31	0.26	**0.32**	0.25

Table 1: For each condition, we report the total number and ratio (from all solutions) of feasible and creative solutions produced. We report both the liberal (novelty threshold $k = 2$) and strict ($k = 3$) settings. Although the empty condition produced the most feasible solutions, other conditions, especially the sentence condition, produced more novel solutions. All inspiration-based conditions produced a higher ratio of feasible and creative solutions than the empty condition.

Metric \ Condition	Purpose	Purp +mech	Purp +mech sent
% creative (k=2) from inspired	0.6	0.57	0.73
% creative (k=2) from non-inspired	0.44	0.47	0.47
% creative (k=3) from inspired	0.42	0.25	0.37
% creative (k=3) from non-inspired	0.17	0.2	0.08

Table 2: Percentage of creative solutions from inspired and non-inspired (=participant indicated this was their own idea) solutions. For all conditions and novelty thresholds, the percentage of creative solutions from inspired-solutions is noticeably higher.

solutions were then tagged with a novelty score (1: the solution is well-known, 2: uncommon solution, 3: a very novel solution). In the following analysis, we treat a solution as either *creative* or *not creative*, by applying a novelty threshold k. We report results for both a liberal setting ($k = 2$) and a strict setting ($k = 3$).

Overall, 3 judges tagged 374 solutions suggested by the participants. Since this is a non-trivial annotation task, the judges were first calibrated over 10 randomly selected solutions. Then, their agreement was computed over another set of 10 solutions. The remaining 354 solutions were randomly split into 3 and annotated separately by the judges. Agreement between the judges was substantial, with 90% full agreement (all judges agreed) on the feasibility scores. Agreement for the novelty score was high as well, with 66% agreement for the liberal case, and 88% agreement for the strict case. For all scores and cases, agreement between at least 2 out of the 3 judges was 100%.

7.1 RQ1: Effect of inspirations

We assess the degree to which each of the inspiration-based conditions (purpose, purpose + mechanism, purpose + mechanism sentence) increased the creativity of the participants. As can be seen in Table 1, participants in the empty condition (as well as the sentence condition) provided the highest absolute number of feasible solutions, perhaps because they did not spend time reading inspirations. However, the *ratio* of feasible ideas (out of all ideas, averaged over all participants) was higher for the inspiration-based conditions, suggesting that **inspirations helped participants produce higher-quality ideas**.

For creativity, we see that for both novelty thresholds $k = 2, k = 3$, the purpose+mechanism sentence condition produced the highest number of creative ideas, but participants in all inspiration-based conditions produced a significantly higher *ratio* of creative ideas compared to participants in the empty condition, again indicating the contribution of the inspirations.

We were surprised to see that the performance of purpose+mechanism was low compared to other inspiration-based conditions. This might indicate that the relation between mechanisms and purposes is not always clear, confusing the participants and increasing the cognitive load, and putting it into a sentence helps participants.

Since **the sentence condition has yielded the best results**, we focus on it, and test its usefulness in producing creative ideas. We run a statistical test to compare the ratios of creative ideas per participant under the empty and sentence conditions. We verify normality and equal variances using the Shapiro-Wilk test and Levene's test, and run Student's t-test to compare the two conditions. The results for the liberal case are significant with $p =$ 0.004, but the results for the strict case were not (p = 0.07), potentially because the number of highly novel ideas was smaller.

For the inspiration-based conditions, we examine the reported source of inspiration per idea. The ratio of creative solutions out of solutions inspired

Metric \ Source	NLI	LLM	Verb
% Feasible solutions	0.84	0.78	0.73
% Creative solutions (k=2)	0.71	0.67	0.54
% Creative solutions (k=3)	0.39	0.4	0.29

Table 3: Percentages of feasible and creative solutions per trajectory type (out of all solutions using this trajectory). The results are calculated over both problems from the experiment. For all metrics and cases, the results for NLI and LLM-based inspirations are higher than those of the verb-based inspirations, indicating a stronger signal for enhancing creative ideation.

by the graph was higher than the ratio of creative solutions out of non-inspired ones (participants' own ideas) across all conditions (Table 2).

7.2 RQ2: Trajectories

To answer our second research question, we compare the usefulness of the inspirations by their trajectory (NLI, LLM, Verb nodes).

We look at the ratio of feasible and creative solutions out of all solutions inspired by a certain trajectory type. The results in Table 3 indicate that both LLM and NLI-based inspirations were able to produce solutions of higher quality than verb-based inspired solutions. This might make sense, as two nodes that only share synonymous verbs might be very far off. To complement this, we observe the percent of inspirations from each source that were used in creative solutions. We find that 38% of the verb-based inspirations were used in very creative solutions (novelty threshold = 3), close to that of the other two sources (44%, 42%). We conclude that **although the LLM and NLI-based inspirations proved superior in our experiment, verb-based inspirations are still useful**.

7.3 Additional insights

Solution generation over time. We compare the number of feasible and creative solutions produced by participants under each condition, as a function of the time from the start of the experiment (Figure 3). Participants under the empty condition produced more feasible (top figure) and creative ideas (the figure depicts the liberal novelty setting, but this is true for the strict setting as well). We hypothesized that participants in the empty condition are not shown inspirations and can immediately start coming up with ideas, leading to better results at the beginning of the experiment. However, this advantage disappears over time, as the subjects in the other conditions exceed the performance of the empty-condition participants. We also note that 33.4% of the participants under the empty condition stated they needed more time to complete the task, as opposed to 50%, 46.7% and 53.3% of the participants under the inspiration-based conditions.

Preliminary exploration of SOTA LLMs. Despite their tremendous popularity, state-of-the-art LLMs struggle with creative thinking and problem solving (Tian et al., 2023; Franceschelli and Musolesi, 2024). We perform a preliminary study to assess the possibility of using LLMs to find creative solutions to everyday problems. We ask GPT-4o (Hurst et al., 2024), a popular SOTA LLM, to generate original solutions to the same problems given to the participants in our experiment. We use similar instructions to those given to participants. Overall, GPT-4o produced 14 solutions (7 for each problem), 8 of which deemed feasible by our annotator. However, further analysis of the solutions showed that *all* solutions already appear online, and some are common, hinting that SOTA LLMs might indeed be limited in producing truly novel solutions.

Satisfaction. After completing the task, participants were asked to complete a short survey. When asked whether similar inspirations would be helpful in the future when tackling a problem, 100% of the participants under the sentence condition answered positively, compared to 75% and 86.7% of participants under the purpose and purpose+mechanism conditions.

8 Related Work

Computational methods aimed at augmenting human creativity and ideation have garnered significant attention. A prominent avenue focuses on leveraging analogy as a powerful cognitive mechanism for generating novel solutions (Gentner, 1983; Hofstadter, 2001). One significant line of work involves the creation and utilization of structured knowledge to identify potential analogies. For instance, the seminal Structure-Mapping Engine (SME) (Falkenhainer et al., 1989) operates on propositional representations.

Harnessing analogies to navigate between ideas has been explored in design-by-analogy works. Recent works (Sarica et al., 2020; Luo et al., 2021) offered to retrieve inspirations from patent data, but focused on semantic similarity, not structural similarity or functional relations. Murphy et al. (2014)

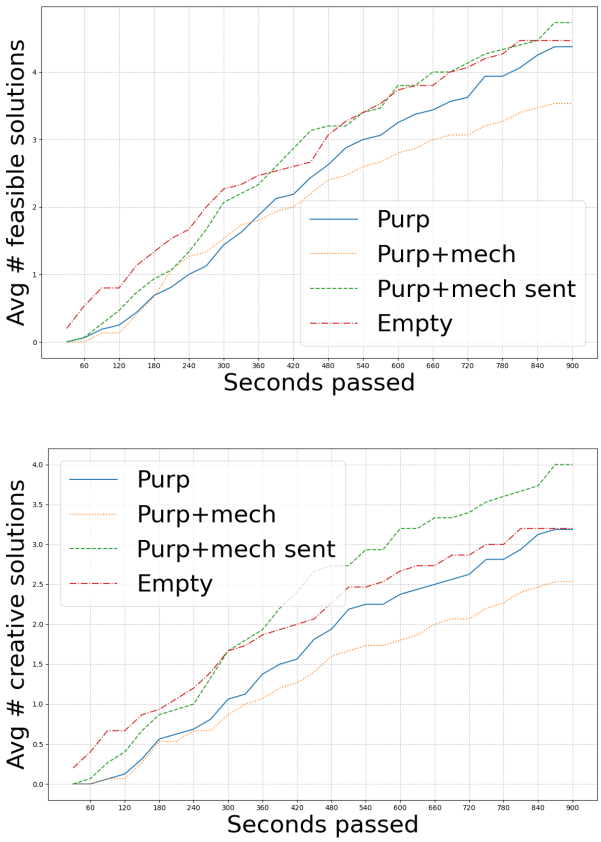

Figure 3: Average number of feasible (top) and novel (bottom) solutions through time passed in the experiment for the different conditions. Participants under the empty condition (green dash-dotted line) produced more feasible and novel ideas at the start of the experiment. As time progressed, participants under the purpose (solid blue) and purpose+mechanism sentence (dashed green) conditions managed to come up with more feasible and novel ideas. This aligns with our hypothesis that generating solutions under the inspiration-based conditions requires additional time.

did try to encode some functionality information, but this was based on shallow keyword embedding, without taking abstraction into account.

More directly relevant to functional thinking are approaches that explicitly encode functional knowledge. The Functional Basis (Hirtz et al., 2002; Stone and Wood, 1999) provides a standardized vocabulary for describing the functions and flows within a system. This framework has been used to develop tools for concept generation, but the vocabularies are often small and restricted, and do not offer the expressivity of our approach.

While analogy and abstraction are still considered hard tasks for machines (Mitchell, 2021), LLMs have shown emergent capabilities of analogical reasoning (Webb et al., 2023; Zhou et al., 2025). This raises exciting possibilities for future work.

9 Conclusions and future work

In this work we propose a method to build Functional Concept Graphs – representations that enable navigation across interconnected functional elements, facilitating abstraction and reframing of problems, and the discovery of analogical inspirations. Unlike previous attempts, our approach can scale to large datasets and results in richer, better-connected and less noisy graphs, whose edges explicitly encode abstraction relations. We also introduce MUSE, an algorithm that, given an FCG and a target problem, produces inspirations that could help users creatively solve the problem.

We demonstrate our method by computing an FCG on 500K patents, which we release for further research. We conduct a user study to evaluate the usefulness of MUSE. Our results indicate that our inspirations resulted in more creative ideas, both in terms of absolute number (up to 19% more creative solutions when using inspirations) and ratio (49% creative ideas without inspirations opposed to 75% in the sentence condition).

In this work, we suggested a simple way of sampling inspirations from the graph. In the future, we plan to explore more sophisticated sampling methods. One immediate option is to sample far-off analogies as inspirations.

Another interesting research direction is using the inspiration graph to enhance creativity in SOTA AI agents. Our initial inspection in Section 7.3 demonstrates that current SOTA LLMs struggle generating truly never-before-seen solutions. We hypothesize that enriching these models in either the training or inference phases would help enhancing their creative problem-solving ability.

We hope our work would inspire further research on enhancing creative ideation by automatically finding structured representations for navigating the design space and finding analogies in large, complex idea repositories.

10 Ethical considerations

Experiment. Our used study was approved by an institutional ethics committee. We do not save any personal information for any of the participants apart from the Prolific ID, which we discard after completing the analysis. Ideas generated by the

subjects remain their own, and we make no commercial use of any of them. Prior to starting the experiment, participants agreed to privacy and data collection terms which were fully described in a consent form.

Usage of AI agents. We did not use any AI agents the writing process of this paper. For coding, we occasionally used Claude 3.7 Sonnet, and verified the output code. The parts of our pipeline that include the usage of LLMs were explicitly described in Section 4.

Reliance on automatic creativity. Our work proposes an automatic assistance to enhance creativity, thus reducing the burden of trying to break the creative fixation for the users. In case our method becomes popular, one might rely on it in creative problem-solving tasks. This raises the potential risk of over-reliance on automatic creativity tools and creating biases.

11 Limitations

In this work, we opted to use patents extracted from the US patents database, and use these patents to draw inspirations from existing solutions. One limitation of our method is that we might miss out on many relevant products and ideas that do not exist in the database. Specifically, users using our solution cannot be inspired by patents not written in English and registered in the US, as they are not part of our dataset. This might introduce a bias and fixate the users to draw analogies from certain types of solutions, ignoring solutions from different cultures.

Similarly, all participants in our user-based study were native English speakers. We did not test how our tool helps ideation for non-native English speakers.

Another limitation of our work is the reliance on CPC tags to collect the mechanism tags. When generalizing to new domains, we would need alternative methods for collecting mechanism tags.

References

Anthropic. 2025. Claude 3.7 sonnet. Accessed: 2025-04-01.

Steven Bird. 2006. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 interactive presentation sessions*, pages 69–72.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library. *arXiv preprint arXiv:2401.08281*.

Brian Falkenhainer, Kenneth D Forbus, and Dedre Gentner. 1989. The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, 41(1):1–63.

Christiane Fellbaum. 2010. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer.

Giorgio Franceschelli and Mirco Musolesi. 2024. On the creativity of large language models. *AI & SOCIETY*, pages 1–11.

Dedre Gentner. 1983. Structure-mapping: A theoretical framework for analogy. *Cognitive science*, 7(2):155–170.

Mary L Gick and Keith J Holyoak. 1980. Analogical problem solving. *Cognitive psychology*, 12(3):306–355.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Julie Hirtz, Robert B Stone, Daniel A McAdams, Simon Szykman, and Kristin L Wood. 2002. A functional basis for engineering design: reconciling and evolving previous efforts. *Research in engineering Design*, 13:65–82.

Douglas R Hofstadter. 2001. Analogy as the core of cognition. *The analogical mind: Perspectives from cognitive science*, pages 499–538.

Keith J Holyoak and Paul Thagard. 1996. *Mental leaps: Analogy in creative thought*. MIT press.

Tom Hope, Joel Chan, Aniket Kittur, and Dafna Shahaf. 2017. Accelerating innovation through analogy mining.

Tom Hope, Ronen Tamari, Hyeonsu Kang, Daniel Hershcovich, Joel Chan, Aniket Kittur, and Dafna Shahaf. 2022. Scaling creative inspiration with fine-grained functional aspects of ideas.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

David G Jansson and Steven M Smith. 1991. Design fixation. *Design studies*, 12(1):3–11.

Moritz Laurer, Wouter Van Atteveldt, Andreu Casas, and Kasper Welbers. 2024. Less annotating, more classifying: Addressing the data scarcity issue of supervised machine learning with deep transfer learning and bert-nli. *Political Analysis*, 32(1):84–100.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jianxi Luo, Serhad Sarica, and Kristin L Wood. 2021. Guiding data-driven design ideation by knowledge distance. *Knowledge-Based Systems*, 218:106873.

James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, volume 5, pages 281–298. University of California press.

Melanie Mitchell. 2021. Abstraction and analogy-making in artificial intelligence. *Annals of the New York Academy of Sciences*, 1505(1):79–101.

Jeremy Murphy, Katherine Fu, Kevin Otto, Maria Yang, Dan Jensen, and Kristin Wood. 2014. Function based design-by-analogy: a functional vector approach to analogical search. *Journal of Mechanical Design*, 136(10):101102.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Prolific. 2014. Prolific. Accessed: 2025-04-10.

A Terry Purcell and John S Gero. 1996. Design and other types of fixation. *Design studies*, 17(4):363–383.

Emily Reif, Daphne Ippolito, Ann Yuan, Andy Coenen, Chris Callison-Burch, and Jason Wei. 2021. A recipe for arbitrary text style transfer with large language models.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Bruce A Reinig, Robert O Briggs, and Jay F Nunamaker. 2007. On the measurement of ideation quality. *Journal of Management Information Systems*, 23(4):143–161.

Serhad Sarica, Jianxi Luo, and Kristin L Wood. 2020. Technet: Technology semantic network based on patent data. *Expert Systems with Applications*, 142:112995.

Robert B Stone and Kristin L Wood. 1999. Development of a functional basis for design. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 19739, pages 261–275. American Society of Mechanical Engineers.

Jiankai Sun, Deepak Ajwani, Patrick K Nicholson, Alessandra Sala, and Srinivasan Parthasarathy. 2017. Breaking cycles in noisy hierarchies. In *Proceedings of the 2017 ACM on Web Science Conference*, pages 151–160.

Yufei Tian, Abhilasha Ravichander, Lianhui Qin, Ronan Le Bras, Raja Marjieh, Nanyun Peng, Yejin Choi, Thomas L Griffiths, and Faeze Brahman. 2023. Macgyver: Are large language models creative problem solvers? *arXiv preprint arXiv:2311.09682*.

Andrew Toole, Christina Jones, and Sarvothaman Madhavan. 2021. Patentsview: An open data platform to advance science and technology policy.

Joe H Ward Jr. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Taylor Webb, Keith J Holyoak, and Hongjing Lu. 2023. Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9):1526–1541.

Ben Zhou, Sarthak Jain, Yi Zhang, Qiang Ning, Shuai Wang, Yassine Benajiba, and Dan Roth. 2025. Self-supervised analogical learning using language models. *arXiv preprint arXiv:2502.00996*.

A Annotated patent example

See figure 4 for a full example of an annotated patent.

B Annotation prompt example

See figure 5 for a prompt and output example for annotating

Patent ID: 10493471

Patent title: Spraying device for a product

Mechanism tags:
- spraying apparatus
- atomising apparatus
- nozzles
- accessories, closures, or fittings
- packages

Purpose tags:
- provide a device for spraying a product
- hairdressing or shaving equipment
- equipment for cosmetics or cosmetic treatments
- containers for storage or transport of articles or materials
- packaging elements

Figure 4: Example of an annotated patent. The patent title is associated with purpose and mechanism tags, demonstrating the problems and solutions offered in it.

C Implementation details

C.1 General implementation details

All code for this project was written in python-3.10. The attached Git repository contains all code, dependencies and commands required to create the inspiration graph, sampling from it, running the experiment and analyzing the results. All packages and datasets used in this work were used solely for academic work, with accordance to their license. All data statistics, model and hyper-parameter choices are described in their corresponding sections. To create the inspiration graph, we used a single A5000 GPU (24GB) for 4 hours. We mostly used it to run the NLI and llama3.1-8b-Instruct models described in section 4.3. In order to speed up the agglomerative clustering process described in section 4.2, we used 8 32GB-CPUs that ran in parallel for 3 days.

C.2 Getting mechanism tags from CPC tags

Prior to training the mechanism classifier described in section 4.1, we process each one of the CPC titles text. first, if it contains more than one text span, we split it so each CPC id may be indicated by multiple titles in our final tags set. This split to multiple text spans may result in a single patent having multiple tags, which are not all necessarily related to it. To deal with this, for each patent we measure the cosine similarity between its title and the CPC tags title's Sentence BERT embeddings, and select only the most relevant one.

C.3 Clustering package and hyper-parameters

For all clustering purposes, we used the algorithms implemented in Scikit-learn (Pedregosa et al., 2011) used under BSD License. **Loose clustering with K-means.** As explained in section 4.3, the loose clustering step is meant to reduce the number of nodes which we aggressively cluster with agglomerative clustering. We choose to use K-means with $K = 10000$.

Aggressive clustering. In order to select the parameters for the agglomerative clustering step, we generate a small dataset of 30 sentences (that did not appear in our data) and split them into clusters. We run the agglomerative clustering algorithm similarity threshold ranging from 0.05 to 0.5, with increment of 0.05. We finally choose similarity threshold = 0.2, since it showed the best tradeoff between the purity (1.0) and NMI (0.97) metrics. We use cosine similarity as the metric for clustering, complete linkage. We use the default values for all other parameters.

C.4 Abstraction with entailment

In order to check whether a problem node entails another, we randomly select a representative purpose tag from each node. We experiment with 2 prefix options – "I want" and "The patent provides". For the NLI model, we use a fine-tuned version of Deberta-V3-large (304M parameters) (He et al., 2021), offered in Laurer et al. (2024). In order to choose the entailment threshold, we tag the clusters generated in section C.3, and tag all abstraction relations between them. We test different values for the entailment threshold t and prefix, settling on $t = 0.5$ and prefix = "I want" since it achieved the highest recall (0.65) and precision (0.9) rates.

C.5 Enhancing connectivity with LLM-based connections and verb-based connections

We extracted verbs using NLTK (Bird, 2006), used under Apache License Version 2.0. As we mentioned in section 4.3, we enhance the connectivity of the graph by creating additional LLM-based nodes created from a set of candidate nodes. We first discuss the process of choosing these nodes. Let h_max be the maximal height of a node in the loose cluster graph. We choose candidate nodes as all nodes whose height is at least $h_{max} - 3$ and distance from the highest node in their path is 2. After selecting the candidate nodes, we use K-means

with $K = 5$ to split these nodes into 5 clusters. We prompt llama3.1-8b-instruct to select (if possible) a subset of the nodes from each cluster that can be abstracted together, and find an abstraction for them. As an additional validation step, for each such cluster c_i, we find the furthest cluster c_j by computing the minimum cosine similarity between all nodes. We add to c_i to 2 nodes which are most dissimilar to nodes from c_j. If any of these nodes were selected alongside original nodes from c_i during the abstraction process, we discard the abstraction.

C.6 Connecting the interim graphs

Similar to the process of enhancing the graph connectivity using LLM-based nodes described in section 4.3, in order to connect the different interim graph we first select a set of candidate nodes for each loose cluster. We select the same candidate nodes as those described in section C.5. In order to connect these nodes with NLI-based nodes, we perform the same process described in section 4.3, over pairs of nodes coming from different candidate node sets. For the LLM-based enhancements, we replicate the same process described in section C.5.

D Inspiration examples

Figure 6 shows an example of 5 inspirations presented in each condition in our experiment.

E Experiment instructions

The full instructions for our experiment are provided in figure 7.

Prompt input and output example for annotating a product description

Prompt:

Example 1:
Method and device for precise invasive procedures.This invention relates generally to the field of invasive medical procedures, and specifically to accurate monitoring of invasive procedures with an imaging system . A method for inserting an invasive tool, including: attaching a frame to a human body adjacent to a portion of the body; acquiring an image of the body; determining a trajectory of the tool on the image; calculating points of intersection between the trajectory and two sheet which are adapted to be inserted into the frame; perforating the sheets at the calculated points; placing the sheet within the frame; and inserting the invasive tool through the perforations.
The purpose of the patent is to monitor invasive procedures.

Example 2:
Hair style device. This invention relates to devices which attach to the hand of a user ,which devices simultaneously blow-dry and style hair on mammals. A hair-styling device containing a hand attachment and a source for heated air under pressure connected to the hand attachment.
The purpose of the patent is to blow dry and style hair

Example 3:
Swivel wheel mount. This invention relates to a child's stroller (a bassinet , a baby buggy or similar device used to support or transport a person) with wheels which swivel. Disclosed is a stroller including a frame member, a swivel mount adapted to receive the frame member, a swivel latch adapted to be received in the swivel mount, a suspension housing including a swivel latch receiving portion, the suspension housing adapted to be attached to the swivel mount, a swivel pin adapted to be received in the frame member, the swivel mount, and the suspension housing, and at least one wheel pivotally attached to the frame member.
The purpose of the patent is to provide a stroller with a swivel wheel

Your input:
Modular engine, such as a jet engine, with a speed reduction gear. The present invention relates to an aircraft propulsion engine, such as a turbojet engine, a multi-flow turbofan, in particular with a high dilution ratio, or a turboprop engine, having a front power transmission shaft, driven by a turbine rotor by means of a speed reduction gear. The present invention relates to an engine (1) with a modular structure comprising a plurality of coaxial modules (A, B, C) with, at one end, a first module (A) comprising a power transmission shaft (3) and a speed reduction gear (7), said power transmission shaft being driven via the speed reduction gear (7) by a turbine shaft (2) secured to one (C) of said coaxial modules that is separate from the first module, the speed reduction gear comprising a drive means (8 and 9) fixed to the turbine shaft (2) and to a journal (13) of a shaft of a low-pressure compressor rotor (1 a), characterized in that it comprises a first nut (16) for fastening the drive means to the journal and a second nut (14) for fastening the drive means to the turbine shaft.
What is the purpose of the patent? What is the context of the patent?

Output example:
The purpose of the patent is to provide a method for reducing the speed of a jet engine.

Figure 5: An example of the prompt we use to annotate patent descriptions with purpose tags, followed by an output example. The in-context prompt consists of 3 examples for annotated patent descriptions, followed by the description to be tagged.

Examples for inspirations provided in the experiment

Condition 1: Purpose

1. Possible inspiration: Think of **a method and apparatus for cooling a work piece**
2. Possible inspiration: Think of **a system for cooling a person**
3. Possible inspiration: Think of **a water cooled door**
4. Possible inspiration: Think of **a computer cooling assembly**
5. Possible inspiration: Think of **a cooling bed system**

Condition 2: Purpose + Mechanism

1. Possible inspiration: Think of **a method and apparatus for cooling a work piece**
   Related concepts:
   - Heat-exchange apparatus
2. Possible inspiration: Think of **a system for cooling a person**
   Related concepts:
   - Air-humidification
3. Possible inspiration: Think of **a water cooled door**
   Related concepts:
   - Combustion engines
4. Possible inspiration: Think of **a computer cooling assembly**
   Related concepts:
   - Vehicle cooling systems
5. Possible inspiration: Think of **a cooling bed system**
   Related concepts:
   - Therapeutic cooling beds

Condition 3: Purpose + Mechanism sentence

1. Possible inspiration: Think of **a method and apparatus for cooling a work piece**
   Related concepts:
   - Heat-exchange apparatus without direct contact enables precise workpiece cooling
2. Possible inspiration: Think of **a system for cooling a person**
   Related concepts:
   - Air-humidification enhances evaporative cooling effects for personal comfort
3. Possible inspiration: Think of **a water cooled door**
   Related concepts:
   - Combustion engines employ water cooling technologies for component protection
4. Possible inspiration: Think of **a computer cooling assembly**
   Related concepts:
   - Vehicle cooling systems inform compact computer cooling assembly design
5. Possible inspiration: Think of **a cooling bed system**
   Related concepts:
   - Medical science applications incorporate therapeutic cooling beds for patient care

Figure 6: Examples for inspirations sampled for the problem "Cool a room". We provide 5 examples for each condition. For clarity, we show the same problem and solution nodes sampled in each condition.

In this experiment we ask you to find **original solutions** to problems.
What do we mean by original solutions? Let's look at an example:

Say the **problem** is 'dry your hair'.

Possible original solutions can be:

1) Existing products that are not meant for drying hair (but could dry hair) such as *leaf blower* (Leaf blowers propel the air and can be used to dry hair.)
2) New inventions. For example, you could think of a special hair spray that helps your hair dry faster, similar to a nail-polish drying spray. You do not need to think through all the details, just the **main idea**.

Known solutions for drying hair, such as a hair dryer or patting the hair with a towel are **not** considered original solutions, and are **not a good answer** for this task.

(This task is for academic study, not for commercial use. Ideas that you generate will remain yours.)

During the experiment we will show you a single **problem**. Then, we will ask you to think of original solutions to that problem (**not the common ways to solve the problem**).

To help you, we provide you with up to **30 potential inspirations**.
The inspirations might help you think of new solutions (but don't worry if they don't, you don't have to use them).
The inspirations might describe problems and concepts that are related to the problem you will be asked to solve. Think of how ways to tackle those problems might inspire you in solving your own problem.
Here is an example of 3 inspirations for the problem of "dry your hair":

1. Possible inspiration: Think of **drying clothes.**
   - Related concepts:
     - Home appliances
     - Rotating tumbler

2. Possible inspiration: Think of **drying nail polish.**
   - Related concepts:
     - Spray

3. Possible inspiration: Think of **washing hair without water.**
   - Related concepts:
     - Baby powder
     - Oil absorption
     - Hair products

You will have **15 minutes** to find solutions.

The output of this task: During this task, you will be asked to fill in a table with (1) your **original solutions**, (2) an **explanation of what you were inspired by**, and (3) the inspiration source id (Inspirations have id numbers assigned to them - the number in **red**). You can select up to 3 inspiration sources for each idea.
In case you didn't use an inspiration source for your solution, select **My idea** in the first inspiration id column .

For example, the answers for the example above would look this:

Solution:	What inspired you:	Inspiration id 1:	Inspiration id 2 (Optional):	Inspiration id 3 (Optional):
A heating hat: a hat with a heating mechanism that you put on top of your head.	Looking on a heating blanket made me think about using the same mechanism to heat the head. This may result in the hair drying faster.	My idea	-----	-----
A hair spray that has drying qualities	I was inspired by a nail polish drying spray: you spray it on your nails and the nail polish dries faster.	2. Drying nail p	-----	-----
Hair drying machine: a machine that spins your hair until it is dry	A dryer spins the clothes to dry them. I thought that maybe a similar idea can work for hair	1. Drying clothe	-----	-----

Please try to think hard and be creative!

Figure 7: The full instructions for our experiment