Formalizing the Unexpected Hanging Paradox in Coq : a New Surprise Definition

Polina Vinogradova[1] [0000−0003−3271−3841]

IOG polina.vinogradova@iohk.io

Abstract. In this work we define a novel approach to formally specifying the unexpected hanging paradox, sometimes called the surprise examination paradox, using a constructive logical framework in the Coq proof assistant. This paradox requires the formalization of the notion of a *surprise* event, which, for the purposes of this paradox, is usually interpreted as the inability to predict what day a specific event takes place. As in existing work, the use of constructive logic allows us to represent knowledge as provability. We make the observation that an inevitable event requires there being strictly more than one possible day on which it can occur, and define surprise accordingly. We specify constraints of the paradox representing this surprise definition. We present a predicate that, in accordance with the paradox constraints, constructs a proposition we can use to reason about the provability of the hanging on a specific day, given that today is a particular day of the week, and the hanging is planned for yet another day. We assert that this offers a satisfying resolution to the paradox in accordance with our intuition. We also provide a comparison to a weaker interpretation of surprise, and an analysis of how it interplays with the use of constructive logic, and compare it to our definition.

Keywords: surprise examination · paradox · unexpected hanging · Coq · constructive logic.

1 Introduction

The unexpected hanging paradox, also known as the surprise examination paradox, is a logical paradox introduced in the the Mind philosophical journal in 1948 [14], and popularized by the Scientific American Mathematical Games column author Martin Gardner, discussed in his work [6]. It describes the notion of a future event that is both certain, and not possible to predict the exact day of occurrence of, formulated as follows :

> A judge tells a condemned prisoner that he will be hanged at noon on one weekday in the following week but that the execution will be a surprise to the prisoner. He will not know the day of the hanging until the executioner knocks on his cell door at noon that day.
>
> Having reflected on his sentence, the prisoner draws the conclusion that he will escape from the hanging. His reasoning is in several parts. He begins by concluding that the "surprise hanging" can't be on Friday, as

> if he hasn't been hanged by Thursday, there is only one day left – and so it won't be a surprise if he's hanged on Friday. Since the judge's sentence stipulated that the hanging would be a surprise to him, he concludes it cannot occur on Friday.
>
> He then reasons that the surprise hanging cannot be on Thursday either, because Friday has already been eliminated and if he hasn't been hanged by Wednesday noon, the hanging must occur on Thursday, making a Thursday hanging not a surprise either. By similar reasoning, he concludes that the hanging can also not occur on Wednesday, Tuesday or Monday. Joyfully he retires to his cell confident that the hanging will not occur at all.
>
> The next week, the executioner knocks on the prisoner's door at noon on Wednesday – which, despite all the above, was an utter surprise to him. Everything the judge said came true.

Existing formalization efforts attempt to address questions like "how can we formally define surprise?", "where is the flaw in the reasoning of the prisoner?" and "was it contradictory for the prisoner to have been hanged on Wednesday?". There is work on tackling these questions in multiple different branches of philosophy and mathematics. An an extensive review of the existing approaches is given in [2].

The definition of surprise as the inability to deduce beforehand the date of the test was first introduced in [16]. A statistical approach to the problem of prediction in this context is discussed in [10]. A proposed solution in the field of epistemology with the use of modal logic, is presented in [7]. An approach involving Kripke semantics, employing the notion of persuasion, is in [9].

The use of non-classical logic, eg., appealing to Goedel's incompleteness, was first applied to the paradox in [5], then in [11], and most recently in [1] [15]. The latter two works assume the underlying logic to itself be constructive, as compared to reasoning via the provability operator Pr to indicate a possibly non-classical statement. This approach aligns most closely with ours. However, in this work we do not analyze the paradox from the point of view of what we can conclude from inconsistency of our system.

The work [8] investigates the relation between the conclusions of the non-classical logic approaches and those from epistemology, as well as presenting four distinct approaches to formalizing the paradox, from which we take inspiration for this work.

In this work, we take the point of view that the constraints of the paradox are fixed and correctly conveyed to the prisoner. Like previous works cited, we chose to represent *knowledge* as constructive propositions, leaving room for uncertainty, or possibility, by dropping the law of excluded middle.

A distinction between existing work and our formalization is that we do not use modal or temporal logic (which is the approach in [7]). Instead, we take advantage of the expressivity of the dependently typed logic of Coq directly to achieve the formalization of the paradox constraints at different points in the week, and parametrized by a different day on which the hanging actually happens.

Because constructive logical is notoriously slippery, we chose to use a proof assistant (Coq, see [3]) to take a closer, more high-assurance look at the interplay between the seemingly simple conditions of this conundrum. There is precedent for the use of proof assistants to tackle philosophical investigation, with the most striking and recent example being a refinement of Kant's categorical imperative [13].

The contributions of this paper are as follows :

(i) A novel definition (in Coq) of surprise that we propose, Section 4, which reflects the following natural language statement : "a hanging has not occurred on or before a given day, and there exist at least two distinct future days on which a hanging is possible". We also give an analysis of how this definition aligns with our intuition;

(ii) The definition of constraints on the prisoner's situation at each point in the week, which specifies the conditions of the paradox and makes use of the surprise definition we propose, Section 6. We also give a formal proof of the impossibility of being surprised by a Friday hanging on Thursday, according to our definition;

(v) The construction of a three-parameter predicate returning a proposition about whether on not a hanging happens on *a given day*, given that we know whether or not a hanging happened on days up and including the parameter day *today*, and a day on which the hanging actually happens, see Section 5, along with a proof that the reasoning specified by this proposition is in accordance with our surprise specification;

(iv) An alternate, weaker formalization of surprise (Section 7). We give analysis of how our definition compares to this one, and why it gives us insufficient reasoning power in the case that the surprise reasoning predicate is non-classical, but in the case that it is classical, allows us to reason ourselves out of the hanging entirely by concluding a sure-thing Friday hanging on Thursday.

For our code, see `https://github.com/polinavino/unexpected_hanging/blob/master/unexpected_hanging.v`.

2 Coq and the Paradox

Coq is a proof assistant that offers a dependently typed formal language. It is capable of verifying formal user-defined proofs of propositions, as well as has support for automation of certain kinds of proofs. The choice of Coq, as opposed to another proof assistant such as Agda, was based largely on the authors' familiarity with the system, as any dependently typed proof verifier that supports constructive logic would serve just as well for the purposes of this formalization.

To formalize the paradox, we need to reason about days of the week on which the paradox could happen, so we begin by constructing a type `weekDay` the terms of which are week days :

```
Inductive weekDay : Type :=
  | monday : weekDay | ... | friday : weekDay.
```

We also define a type `weekAndBefore`, which represents all the weekdays in the type above, plus the Sunday that comes before — the purpose of this type is to represent all the days on which one can consider the possibility of surprise, differentiating it from the subset of days on which the hanging can occur. We also define comparison function `<`, which computes whether a given `td` is before `d`, following real-life weekday logic, eg. Sunday is before Monday.

It is important to emphasize here that $=$, $\geq$, $<$ are all *decidable* comparison functions on days — purely as a consequence of considering weekdays as totally, decidably ordered numbers, even in constructive logic. Any predicates formulated solely out of those comparisons together with logical connectives are also decidable, with the implication that provability, knowledge, and truth are the same for such predicates, leaving no room for uncertainty. For this reason, solutions of the paradox constructed out of only such decidable predicates (eg. [11]) are operating in classical logic.

Next, we can specify the type of a function that, given a day of the week, returns True if we can prove the hanging happened then, and False if we can prove it did not. We reason about this function in the presence of preconditions that are formal interpretations of those described in the paradox. We leave this function as a variable :

```
Variable hangingOnDay : weekDay → Prop.
```

We discuss the existence of such a predicate, and how it changes depending on what day today is, and when a hanging actually happens, in 5. Next, we define a predicate that formalizes the notion that no hanging has occurred yet (up to and including the parameter day `td`, representing *today*) :

```
Definition noHangingYet (td : weekAndBefore) :=
  ∀ d, td ≥ d → ¬ hangingOnDay d.
```

This says that for any day `d`, if it is before today `td`, no hanging happened on `d`. We use the double negation `¬¬ hangingOnDay d` to formalize the statement that it is not possible to disprove that a hanging occurs on day `d`. That is, a hanging is *possible* on a given day.

3 Uniqueness of hanging.

Uniqueness of the hanging day plays an important role in the definition of surprise. We define the predicate `uniqueHanging` formalizing that after a given day `td`, there can be at most one day on which a hanging occurs. So, `uniqueHanging dayBefore` states this about the entire week.

```
Definition uniqueHanging (td : weekAndBefore) :=
  ∀ d d', td < d ∧ td < d' →
  hangingOnDay d → hangingOnDay d' → d = d'.
```

The constraint, `uniqueHanging`, that a (provable) hanging day is unique is, in fact, equivalent to the constraint that a (possible) hanging day is unique. Moreover, negation of uniqueness is implied by a stronger statement (`twoPossible`), which explicitly requires the presence of at least two possibilities for the hanging date. Note here also that this reasoning does not rely on any additional knowledge about the hanging predicate.

```
Definition uniqueMaybe (td : weekAndBefore) :=
  ∀ d d',
  td < d ∧ td < d' →
  ¬ ¬ hangingOnDay d →
  ¬ ¬ hangingOnDay d' →
  d = d'.

Lemma uniqueMaybeEqv (td : weekAndBefore) :
  uniqueHanging td ↔ uniqueMaybe td.

Definition twoPossible (td : weekAndBefore) :=
  ∃ d d', td < d ∧ td < d' ∧ d ≠ d'
  ∧ ¬¬ hangingOnDay d ∧ ¬¬ hangingOnDay d'.

Lemma twoNotUnique : ∀ td,
  twoPossible td → ¬ uniqueHanging td.
```

We make an observations about this : a future hanging is necessarily not unique. This seems wrong — we would expect a hanging to be unique, it's implicit in the description of the paradox. However, we define surprise in part as a non-uniqueness of a future hanging.

4 A lack of surprise

Surprise is difficult to define, so we try to define, instead, what it means to be certain about when a hanging happens, given a collection of days on which it can happen. We define what it means for us to know that a hanging happened before today `td`,

```
Definition knowHanging (td : weekAndBefore) :=
   (∃ d, td ≥ d ∧ hangingOnDay d) ∧ (uniqueHanging dayBefore).
```

For this predicate to be true, the function `hangingOnDay` must be provably ↔ `True` for exactly one day `d` of the entire week, `False` for all other weekdays, and this day `d` is before or on `td`. This predicate `knowHanging` should hold exactly when a hanging has already happened before today.

Now, let us consider the negation of these two conditions for days `d` after `td` :

```
Definition dontKnowHanging (td : weekAndBefore) :=
  ¬ ((∃ d, td < d ∧ hangingOnDay d)
    ∧ (uniqueHanging dayBefore)).
```

So, either there is no hanging, or it is not unique. This represents surprise fairly well, however, it allows for the possibility that no hanging happens at all in the rest of the week, which should only be true if one had occurred before `td`. So, we must add to this definition a constraint that if we are able to prove that no hanging happens this week, we have proven `False` (we will handle the case for when a hanging did happen already separately).

Note here that while we want to say that on each future day, proving a hanging occurrence should not be possible, adding this to our definition will allow us to prove that no hanging happens at all — and we want the opposite of that. That is, the proposition that no hanging happens the entire week implies that no hanging is even possible, as `¬ hangingOnDay d → ¬ (¬ ¬ hangingOnDay d)`. We notice that the both `uniqueHanging td` and its negation are trivially satisfied whenever `¬ (∃ d, td < d ∧ hangingOnDay d)`. For this reason we use the stronger `twoPossible td` in our definition of surprise, which contradicts the possibility of there not being a hanging at all, and guarantees two possible days.

So, regardless of when the hanging actually happens, we define what it means for surprise to being possible after `td` as,

```
Definition surprise (td : weekAndBefore) :=
  (noHangingYet td) ∧ (twoPossible td).
```

5 The hanging predicate

According to the definition of the paradox, a Wednesday hanging satisfies the constraints. However, the spirit of the paradox seems to suggest there is nothing special about a Wednesday hanging. Here we explain how to parametrize our reasoning by different possible hanging days. Recall that we want to reason about what sort of function `hangingOnDay` satisfies the constraints of the paradox.

We already parametrize the function `surprise`, representing our inability to predict a hanging day (ie. choose a unique hanging day) after today `td`, by what day today `td` is. This function, in turn, builds propositions to reason about whether a hanging day happened on some day `d` by using the predicate `hangingOnDay`. We note that this predicate must actually be a different predicate depending on what day `td` is as well, since we must vary it to indicate that we have full knowledge of all days before today. We require one more additional in our hanging predicate — the actual day of the occurrence of the hanging, which may be Wednesday or otherwise. So, all in all, a function

`hangingOnTodayIsReasoningAbout hang td d : Prop`

that we will define so that it satisfies the paradox constraints, has three parameters :

(i) `hang : weekDay`, which is the day on which the hanging actually occurs (once today is or or after this day, surprise should no longer be possible, but the paradox conditions may not be violated)
(ii) `td : weekAndBefore`, which is the day that is "today", ie. the day on which the prisoner is reasoning about the hanging (all the days before and including today should have a classical predicate about whether a hanging occurred or not)
(iii) `d : weekDay`, the day about which the prisoner is reasoning (eg. tomorrow), but should not be able to prove or disprove the occurrence of a hanging on, unless it already happened prior to `td`

Parametrizing by hanging day lets us define a set of reasoning predicates each of which assumes a different actual day of hanging. We reformulate the reasoning predicate to be parametrized in this way, To accommodate this change, we also parametrize all the predicates used in the definition, ie. `noHangingYetparam hangingOn hang td`, etc.

6 Paradox constraints

We now present the constraints to be placed by the description of the paradox on the hanging predicate `hangingOnTodayIsReasoningAbout` discussed above.

```
  Definition twoPossiblePRDXparam
      (hangingOn : weekDay → weekAndBefore → weekDay → Prop)
      (hang : weekDay) (td : weekAndBefore) :=
    (td ≥ hang ∧ (hangingOn hang td hang)
       ∧ uniqueHangingparam (hangingOn hang td) dayBefore)
    ∨
    (td < hang ∧ noHangingYetparam td
       ∧ twoPossibleparam (hangingOn hang td) td).
```

We want to define a `hangingOnTodayIsReasoningAbout` such that

```
Lemma hangingFuncOk : ∀ hang td,
    ¬ (td = (someWeekDay thursday) ∧ hang = friday) →
    twoPossiblePRDXparam
        (hangingOnTodayIsReasoningAbout hang td) td.
```

We explicitly exclude being surprised on Thursday by a Friday hanging, as it appears to legitimately be a situation devoid of surprise (see discussion below). The statement of the paradox can be interpreted using these constraints and the predicate definition as :

> The executioner plans the hanging to be on the parameter day `hang`. He tells the prisoner that a correct approach to reasoning about whether the hanging happens on a given day, `hangingOnTodayIsReasoningAbout`, must satisfy the constraints of the paradox, as specified by `twoPossiblePRDXparam`.

We define the predicate :

```
Definition hangingOnTodayIsReasoningAbout hang td d : Prop :=
  (td ≥ hang → hang = d) ∧ (td < hang ∧ td > d → False)
  ∧ (td < hang ∧ td < d → (¬ ¬ hf hang td d)).
```

which says that if today is after the day of the hanging, this is a provable predicate when the day being reasoned about is the same as the hanging day. If today is prior to the hanging, and we are reasoning about a day before `td`, the day we reason about cannot have a hanging. Finally, if the hanging day is not yet reached, a hanging occurrence cannot be disproved about any future day.

Next, we observe that the value `True` instead of `hf hang td d` would actually do the trick. In fact, we prove that it does by defining it that way and proving lemma `hangingFuncOk`. However, claiming that a hanging is going to happen on all future days, and we can prove it, does not sound like a satisfying resolution. But we argue that it may well be — since in order for us to "know when the hanging happens", we must prove that it happens on exactly one day. If we prove that it happens on multiple days in the future, this does not constitute knowledge of when the one promised hanging will happen.

We do not specify a non-classical predicate `hf` as Coq does not support straightforward definition of non-terminating functions. We conjecture (and aspire to show as part of future work) that it is possible to define `hf` to be strictly weaker than `True` for the relevant inputs, satisfying the paradox. Such a definition would align more closely with the intuition that all future days should have a *possibility* of hanging.

The first thing we can formally conclude about this definition is that it indeed rules out a Friday hanging. The following lemma says that it is not possible that by Thursday, no hanging has happened, but there are still two distinct possible days on which it can happen in the future.

```
Lemma cantBeSurpFriday hangingOnTodayIsReasoningAbout :
  ∀ hang,
  twoPossiblePRDXparam hangingOnTodayIsReasoningAbout hang
    (someWeekDay thursday)
  → noHangingYetparam hangingOnTodayIsReasoningAbout hang
    (someWeekDay thursday)
  → False.
```

The proof (see code) is simple, since there is only one day (Friday) left in the week, and no hangings are possible on past days. Inductive reasoning in attempt to conclude that a Thursday hanging is predictable on Wednesday does not work because we do not have enough data on Wednesday to conclude whether a hanging will be Thursday or Friday. That is, to discount Friday as a possibility of surprise hanging, we *need to know that there was no hanging Thursday.*

7 At least one possible day

.

Surprise requires that a future hanging is possible — on more than zero of the remaining weekdays after today. There is precedent [8] for defining surprise in a way that allows a Friday hanging to be a surprise in a consistent way. We specify this interpretation of surprise, together with the other the conditions of the paradox (ie. that once a hanging occurs, there will be no future hangings) :

```
Definition onePossiblePRDXparam
     (hangingOn : weekDay → weekAndBefore → weekDay → Prop)
     (hang : weekDay) (td : weekAndBefore) :=
  (td ≥ hang ∧ (hangingOn hang td hang)
      ∧ uniqueHangingparam (hangingOn hang td) dayBefore)
  ∨
  (td < hang ∧ noHangingYetparam td ∧
      ∃ d, td < d ∧ ¬ ¬  (hangingOn hang td) d).
```

where the first disjunct is the same as before, and the second one corresponds to "today is not yet Friday, and there is a possible day on which a hanging may happen", which we refer to as the `onePossible` definition of surprise. This is a strictly weaker definition than `twoPossiblePRDXparam`, as the last conjunct of the disjunct requires only one possible day to exist, rather than two distinct ones. So, the same predicate `hangingOnTodayIsReasoningAbout` satisfies these constraints as well.

No inconsistency is introduced here, in fact, the hanging can still be a surprise even if it happens on a Friday! The intuition behind this is : if no hanging happened by Thursday, it is still only possible to prove `¬ ¬ hangingOn hang td friday`, from which we are not necessarily able to deduce that `hangingOn hang td friday`. However, defining `hf hang td d = True` for all future days (when there is no hanging yet) immediately results is being able to prove a unique (future) Friday hanging when today is Thursday.

Let us consider the possible, ie. ¬ ¬ case, again, and what happens if we impose an additional constraint stating that exactly one *possible* hanging day implies that it *provably happens* on a specific day. Similar reasoning is also explored in [8], with a similar conclusion to the one we draw here. The following proposition states that there is a possible hanging day, and that uniqueness of hanging day possibility implies certainty of hanging on that day :

```
Definition existsUniqueHappens :=
  (∃ d, ¬ ¬ hangingOnDay d)
  ∧
  (∀ d d', ¬ ¬ hangingOnDay d
    → ¬ ¬ hangingOnDay d' → d = d')
  → ∃ d, hangingOnDay d.
```

Now, the following statement expresses that `existsUniqueHappens` lets us conclude that `hangingOnDay` must then be classical (the proof is in the associated code) :

```
Lemma euhImpClassical :
  (uniqueHanging dayBefore) →
```

```
(∃ d, ¬ ¬  hangingOnDay d) →
existsUniqueHappens →
(∀ d, ¬  hangingOnDay d ∨ hangingOnDay d).
```

This is the crux of the reasoning the prisoner engages in to informally to arrive at the judgement that if a hanging hasn't happened by Thursday, it must happen on Friday. Note here that the inductive reasoning in the prisoner uses to conclude that the hanging cannot ever be a surprise is, in some sense, superfluous — we can use constructive logic reasoning to prove, without induction, that "if we can conclude from existence plus uniqueness of a possible hanging day, that it is certain on that day, *our judgement about hanging occurring on any day must necessarily be classical*". The proof makes use of the fact that the equality comparison `d = d'` is classical. This definition leaves us with the following conclusions about defining surprise as having at least one possible hanging day :

(i) a definition of surprise with a non-`True` `hf` function may not be strong enough to allow us to conclude `existsUniqueHappens`, in particular, that a hanging must happen Friday given that it has not occurred by Thursday, and is therefore not a surprise in that case ; and

(ii) if we *were* to be able to conclude `existsUniqueHappens`, reasoning about surprise becomes classical, so we can always figure out the unique hanging day at any point in the week.

Both possibilities appear problematic : (i) does not allow us to make a conclusion that we would like make according to intuition, and (ii) removes any ambiguity about the future hanging day, and therefore, the possibility of surprise.

With the two-possible definition of the paradox, surprise by Friday hanging is anyways not possible on Thursday. Our definition, however, does not allow us to conclude directly that a hanging will definitely happen on Friday once we get to Thursday. Rather, it is that the conditions for surprise are not satisfied on Thursday in another way (insufficient possible days). An impossibility of a Friday hanging being a surprise on Thursday has no effect on the rest of the week (as shown in 6) in the sense that the surprise constraints are satisfied on other combinations of days for all `d`.

Note here that the difference between one- and two-possible versions of surprise, with respect to having proofs of future hangings, is that assuming (`hf friday (someWeekDay thursday) friday = True`), rather than a only possibility of a hanging on Friday, breaks the one-possible paradox case. This is because the one-case, assuming the `True` version of `hf`, allows a unique Friday hanging to be proved on Thursday without contradiction, but the two-case does not allow this. However, using a weaker `hf` in the one-case, for which we can only prove a hanging possibility, does not allow us to conclude that the hanging will definitely happen on Friday even when it is the only possible day. So, we must *rely* on the reasoning in `existsUniqueHappens` to conclude surprise is not possible on Thursday, which leads to the conclusion that the prisoner arrived at. In the two-case, we can use the paradox definition to conclude that a Friday hanging will not be a surprise on Thursday, and can avoid requiring this reasoning.

8 Conclusion and Future Work

We have presented a formalization of the constraints of the unexpected hanging paradox parametrized by the day the hanging is planned, together with a family of predicates (a distinct one for each day the hanging will actually take place) that lets us reason, according to the knowledge the prisoner has, about a hanging happening or not happening on past and future hanging days from the perspective of each of the days of the week. This formalization appears to capture the conditions of the paradox and align with our intuition. We formalized this definition, along with some related proofs, in the Coq proof assistant, relying on the axioms of constructive logic to represent knowledge, and the possibility of uncertainty. A few key ideas were needed to achieve this.

First, we observe that knowing when a hanging happens requires that we are able to reason that a hanging happens on some day of the week, and that this day is unique. We define a lack of knowledge of when a unique hanging happens (ie. the possibility of surprise) as the negation of this, together with the assumption that there exists day on which a hanging is possible. This approach to defining surprise allows us to exclude Thursday as a day on which we can be surprised by a Friday hanging, but include all other combinations of hanging day, today, and day about which we reason, as meeting the paradox conditions. However, we notice that our definition of hanging does not require us to adhere to non-classical reasoning if all that is needed to claim uncertainty about the hanging day is the ability to prove that it happens on multiple different days, rather than one unique one.

We compare our definition to a formalization of paradox constraints with a slightly weaker definition of surprise. This definition requires at least one day to exist in the future on which we cannot disprove the hanging. We notice that a classical definition of the hanging predicate leads to the ability to prove knowledge of a Friday hanging on Thursday, without contradiction. However, a non-classical definition of the hanging predicate does not allow us to definitively conclude a Friday hanging on Thursday — only the possibility of it. Imposing this conclusion results in the ability to prove the existence of a unique future hanging, thus contradicting our notion of surprise.

As part of future work, we conjecture this paradox formalization could be further analyzed by way of considering its relationship to the axiom of choice. This is due to its (at least surface level) resemblance to the way the AC makes a connection between classical logic and a choice function [4] as well as arbitrary elements [12].

Another future direction we consider is generalizing the surprise hanging approach we propose to using constructive logic for describing and proving the existence of a function `myPick` which represents choosing an arbitrary value from a decidable set. In particular, given a decidable set `W`,

> For any subset `S` $\subseteq$ `W` of cardinality at least 2, such that for all `s` $\in$ `W - S`, $\neg$`myPick s`, there exist at least two distinct elements in `S`, such that $\neg\neg$ `myPick s`.

References

1. Ardeshir, M., Ramezanian, R.: A solution to the surprise exam paradox in constructive mathematics. The Review of Symbolic Logic **5**, 1 – 8 (12 2012). https://doi.org/10.1017/S1755020312000160
2. Chow, T.Y.: The surprise examination or unexpected hanging paradox (1999). https://doi.org/10.48550/ARXIV.MATH/9903160, https://arxiv.org/abs/math/9903160
3. CNRS, contributors: Coq reference manual (2021), https://coq.inria.fr/distrib/current/refman/
4. Diaconescu, R.: Axiom of choice and complementation. Proceedings of the American Mathematical Society **51**(1), 176–178 (1975), http://www.jstor.org/stable/2039868
5. Fitch, F.B.: A goedelized formulation of the prediction paradox. American Philosophical Quarterly **1**(2), 161–164 (1964), http://www.jstor.org/stable/20009132
6. Gardner, M.: Unexpected Hanging Paradox and Other Mathematical Diversions. University of Chicago Press (1991)
7. Halcrow, W., Holliday, W.: Simplifying the surprise exam (2016)
8. Halpern, J.Y., Moses, Y.: Taken by surprise: The paradox of the surprise test revisited. Journal of Philosophical Logic **15**(3), 281–304 (1986), http://www.jstor.org/stable/30226356
9. Harrison, C.: The unanticipated examination in view of kripke's semantics for modal logic (1969)
10. Kim, B., Vasudevan, A.: How to expect a surprising exam. Synthese **194**, 3101–3133 (2017)
11. Kritchman, S., Raz, R.: The surprise examination paradox and the second incompleteness theorem (2010). https://doi.org/10.48550/ARXIV.1011.4974, https://arxiv.org/abs/1011.4974
12. van Lambalgen, M.: Independence, randomness and the axiom of choice. The Journal of Symbolic Logic **57**(4), 1274–1304 (1992), http://www.jstor.org/stable/2275368
13. Lindner, F., Bentzen, M.M.: A formalization of kant's second formulation of the categorical imperative (2018). https://doi.org/10.48550/ARXIV.1801.03160, https://arxiv.org/abs/1801.03160
14. O'CONNOR, D.J.: PRAGMATIC PARADOXES. Mind **LVII**(227), 358–359 (07 1948). https://doi.org/10.1093/mind/LVII.227.358, https://doi.org/10.1093/mind/LVII.227.358
15. Ramezanian, R.: A constructive epistemic logic with public announcement (non-predetermined possibilities). CoRR **abs/1302.0975** (2013), http://arxiv.org/abs/1302.0975
16. Shaw, R.: The paradox of the unexpected examination. Mind **67**(267), 382–384 (1958). https://doi.org/10.1093/mind/lxvii.267.382