Proving Vizing's Theorem with Rodin

Joachim Breitner

March 25, 2011

Proofs for Vizing's Theorem tend to be unwieldy unless presented in form a constructive algorithm with a proof of its correctness and termination. We implemented such an algorithm in the modelling formalism Event-B and performed a machine-checked correctness proof with the Rodin tool.

Contents

1. Vizing's Theorem

In this project, we proved Vizing's Theorem:

> For a finite undirected graph without autoloops and without multiple edges, at any vertex of which no more than N edges meet, $N+1$ colours suffice for an edge coloring such that edges incident on the same vertex are of different color.

This theorem, although proved earlier as well, was considered by Rao and Dijkstra in 1990, who gave an easier to understand proof by describing a constructive algorithm that colors the edges correctly, along with a proof that this algorithm terminates and is correct [RD90]. This approach was simplified and streamlined by Misra and Gries [MG90]. They also introduce names such as "fan" and "*cd*-path" for concepts occurring in the constructions. Their algorithm consists of consecutively executing these steps until all edges are colored:

> Let $X \mapsto Y$ be an uncolored edge.
> Let $\langle Y \dots l \rangle$ be a maximal fan.
> Let c be a color that is free on X, and d a color that is free on l.
> Invert the *cd*-path.
> Choose w such that $\langle Y \dots w \rangle$ is a prefix of the fan $\langle Y \dots l \rangle$ and d is free on w.
> Rotate the fan $\langle Y \dots w \rangle$ and color the edge $X \mapsto w$ with the color d.

A *fan* is a list of distinct neighbours of X, starting at Y, such that the edge $X \mapsto fan(k+1)$ is coloured and this color is free on the node $fan(k)$. The colors on such a fan can be rotated, e.g. moved from the edge $X \mapsto fan(k+1)$ to $X \mapsto fan(k)$ without invalidating the coloring.

The *cd*-path is the largest sequence of nodes starting at node X and following the edges colored c or d. The inversion of such a path changes the colors on these edges from c to d and vica-versa. This is also a transformation that preserves validness of a coloring.

The proof given in the Misra and Gries paper establishes that the operations above indeed preserve the validness of the coloring. Furthermore, they prove that after the inversion of the *cd*-path, a prefix of the path with the property given in the algorithm exists. This is done by case analysis: If no fan edge has color d, the *cd*-path was empty to begin with. If there was a fan edge with color d, then d is free on the preceding vertex, say v, on the fan. Either v is in the *cd*-path, then the original fan suffices. Or v is no in the *cd*-path, in which case the prefix $\langle Y \dots v \rangle$ is a suitable fan.

In any case, the inversion of the *cd*-paths frees the color d on X, which is preserved by rotation. After rotation, d is free on X and the last edge of the fan w, so this edge can be colored.

For more details on the proof, refer to the implementation below or to the original paper.

2. Event-B and Rodin

Event-B is a formalism conceived by Jean-Raymond Abrial to model and verify systems. The main characteristic of Event-B is the concept of refinement: An abstract specification is refined by a slightly more concrete model, which is proved correct with regard to the specification. Then this model is again refined by something more concrete, and again the correctness of the new model is proved, but not against the original specification, but only against the previous model. This is iterated until the final, concrete implementation is reached. By transitivity of implication, the final model is known to correctly implement the specification.

Another important characteristic of Event-B is the use of events, which consist of logical predicates as guards, deciding when an event may occur, and (possibly non-deterministic) actions, modifying the state of the model. The expected behaviour of the model is formalized by predicates called invariants, which have to be provably preserved by every event.

Rodin is an Eclipse based platform to perform modelling within this framework. It allows to define the models, calculates the proof obligations and either solves them automatically using external theorem provers or gives the user to manually perform the proof. It is extensible with plugins, for example to generate LaTeX documents describing the models (as used in this paper) or to generate additional proof obligations such as the absence of deadlocks.

3. Proof validation

A Rodin model is an unusual way of presenting a proof for a mathematical statement. So the question arises whether it is a valid one? It is sound given the following assumptions are made or independently verified.

- Rodin is sound, e.g. it would reject any invalid automatic or manual proof. This is actually a strong assumption. Previous versions of Rodin have contained soundness errors[1],and the Release Notes[2] for new versions contain a disclaimer:

  > However, despite the total commitment of our teams to insure the soundness of the platform, some unexpected and unknown soundness issues could remain.

  In the case of Vizing's Theorem, the result is already well established. If this were some new result, relying only on the Rodin platform for verifying the proof obligations would not constitute a rigorous proof.
- Rodin correctly generates all proof obligations required to ensure that an executable algorithm in the final refinement indeed fulfills the specification given in he first refinement. Given that this is the core idea of the Event-B formalism, its theory and implementation is likely to be thoroughly reviewed.
- The formalisation of the graph and the guard of the finish event in the first refinement are faithful representations of the assumptions and conclusions of the theorem to be proved.
- Every event introduced in later refinements is, at one stage, shown to be convergent.

4. Refinement strategy

Event-B is event based, so imperative algorithm cannot be reasoned about directly. The usual approach here is to have one particular event, here called *finish*, which indicates the end of the

[1] `http://sourceforge.net/tracker/?func=detail&aid=3158594&group_id=108850&atid=651669`
[2] `http://wiki.event-b.org/index.php/Rodin_Platform_2.1.1_Release_Notes`

Model	**POs**	**auto.**	**manual**
VizingTheorem	305	250	55
Input	2	2	0
m00 Spec	5	5	0
m01 Colorops	15	8	7
m02 FixX	18	13	5
m03 FixY	18	14	4
m04 Fan	65	54	11
m05 CDpath as path	12	12	0
m06 CDpath building	80	64	16
m07 Event ordering	50	47	3
m08 Conv Fan Build	13	9	4
m09 Conv CDpath	11	10	1
m10 Conv Fan Color	9	6	3
m11 Conv Stages	6	6	0
m12 Deadlock Freedom	1	0	1

Table 1: Proof obligation statistics

algorithm. In the very first model, the guard of this event is set to the desired result of the algorithm. In further refinements new events are added as required to reach this goal. Each of these events is eventually marked as *convergent*, thus ensuring that the program does not run forever. If additionally deadlock freedom is proved for the very last model, this implies that eventually, the *finish* event will be executed. At this point, we know that the program is correct and terminates.

Our construction of the coloring algorithm consists of a context, which defines the constants of the problem (the vertexes, the graph and the available colors) together with our assumptions about them (represented as axioms of the system). An initial model gives the abstract and non-executable specification of our problem, followed by 12 refinements. The last five of these do not modify the model; their purpose is to house the convergence and deadlock freedom proofs.

The following list gives a quick overview of the refinements and lists when each event occurs the first time. Table 1 gives an overview of the number of proof obligations in each refinement, and how many of them had to be done manually.

- Input: Defines the input of the problem with the given assumptions.
- m00 Spec: Abstract specification of the algorithm.
  New events: *INITIALISATION* and *finish*.
- m01 Colorops: Defines the possible operations on the coloring of the graph and establishes their correctness. Includes the proof of the convergence of *color1*.
  New events: *color1*, *colormove*, *invertpath*
- m02 FixX: Introduces the vertex variable `X` and modifies the events to only work with that vertex, and update it when appropriate.

- m03 FixY: Introduces the vertex variable `Y` and modifies the events to only work with that vertex, and update it when appropriate.
- m04 Fan: Introduction of the concept of a fan.
  New events: *extend_fan*, *fan_done*. The event *color1* is split into *color1a* and *color1b*.
- m05 CDpath as path: The set representation of the *cd*-path is refined to refer to the image of a path, as required by the algorithm.
- m06 CDpath building: The *cd*-path is represented in a variable, the calculation of the *cd*-path is implemented.
  New events: *extendcdpath*, *noinvertpath*.
- m07 Event ordering: The event guards are modified to ensure that they run in the desired order. To this end, a variable `stage` is introduced.
- m08 Conv Fan Build: Proofs convergence of stage 3, the construction of the fan.
- m09 Conv CDpath: Proofs convergence of stage 2, the construction and inversion of the *cd*-path..
- m10 Conv Fan Color: Proofs convergence of stage 1, the re-coloring of edges on the fan.
- m11 Conv Stages: Completes convergence across all stages.
- m12 Deadlock Freedom: Proofs the deadlock freedom.

5. Rodin impressions

Rodin is a modelling tool, and not a theorem prover. This is one conclusion of this project. Although we were able to obtain the proof we wanted, and assuming soundness of Rodin, the proof is a valid one, there were obstacles in the way. We will discuss some of the difficulties encountered. Note that we used Rodin for the first time and without guidance, some of the difficulties might be due to ignorance on our side and not limitations of the framework.

There is no support to define custom predicates, i.e. introduce abbreviations for formulas, in Rodin itself. A plugin is available, the Theory plugin, to extend Rodin's mathematical notation, but it only allows to define predicates on a global level and not with regard to user-introduced axioms or even within the scope of one machine. As a work-around, we defined the axiom *valid* in the context, which is the set of all valid colorings, and used $c \in \mathit{valid}$ to express that the predicate is true for the coloring c. This works, although it can become annoying as we had to manually unroll this definition to access the properties of a valid coloring. In hindsight it might have been easier if we had, at least for the omnipresent variable `coloring2`, given these properties directly as invariants.

Rodin has support to express statements about the cardinality of finite sets, but the only few proof rules are available. Therefore, to transform

$$\mathrm{card}(A \setminus B) = \mathrm{card}(A) - \mathrm{card}(B),$$

where we know $B \subseteq A$, we had to first express it in terms of cardinality of a union to be able to use the existing rule

$$\text{card}(A \cup B) = \text{card}(A) + \text{card}(B) - \text{card}(A \cap B)$$

and continue from there.[3]

Pulling the cardinality operator across a function, e.g.

$$\text{card}(\text{ran}(f)) \leq \text{card}(\text{dom}(f)),$$

or the equality of domain and range of injective functions were not expressible at all (or we were not able to find a workaround). This is the reason for the introduction of `axm13`, a fact that should be provable given the other axioms.

Occasionally, we were missing an indicator function, at least for natural numbers. We worked around it by using expressions such as $(min(\{stage, 2\}) - 1)$, which worked well enough. Although Rodin has rules to automatically replace an expression $min(\{3, 2\})$ by 2, it would leave an expression $3 - 2$ in the goal. To solve this, the user has to add $3 - 2 = 1$ as a hypotheses, prove this using the automatic provers, and then manually apply the quality.

In the action specification of an event, it is not possible to refer to the after-state x' of a variable x set before. This causes repetition, for example in the event *color1* of the model FixY: In `act1`, we update the variable *coloring2* with some larger expression and have to repeat that expression in `act2`. Similarly, the non-deterministic assignment to X and Y in `act3` of the initialisation causes a proof obligation of feasibility. Later refinements have to assign the variables *fan* and *cdpath* in a manner that deterministically depends on the new values of X and Y. But as the assignment cannot refer to X' and Y', so the action `act3` has to be modified to also set *fan* and *cdpath*. This causes the same feasibility proof obligation to reoccur in later models.

In general, proofs within Rodin are write-once, read never. This is very different from, for example, the structured proofs done with the theorem prover Isabelle[isa], and discourages large proofs. We assume that this is intentional, as a very large proofs probably indicate that some refinement step was too large, and that using additional events or invariants, the proof would be smaller. Nevertheless, some large proofs were not avoidable in the course of this project.

Although Rodin seems to try hard to retain proofs even if the underlying model is changed, it does not always succeed and would throw away the old proof then. Given the time spent on some proofs, this stopped us from doing more clean-up of invariants and events after finishing the modelling. Even a simple change, such as removing the unnecessary variable `coloring` of the initial model would lose many proofs in later models.

There are few choices to extract the models in a presentable way. Basically, there is only the LaTeX plugin, which is used in this report. It can export one model at a time, which results in a lot of clicks when exporting the whole development, and produces full LaTeX documents, which were to be mangled by bash and perl to obtain fragments that could be included in this document. Because of the single-model focus, it repeats guards and actions of refined events even when the event is only extended.

[3]Upcoming versions of Rodin are likely to have a rule for the cardinality of set differences.

6. Lessions learned

If we would do the same project again, we would tackle it with slight modifications. Some of these changes should be possible to implement after finishing the project as well, if it were not for the problem mentioned above that some changes cause too many proof obligations to be lost.

The *finish* event of the first refinement only needs a guard. As our goal is to proof a theorem, and not to develop a real-life system, we are not so much interested in the final coloring but only in its existence. Therefore, the guard

`grd1 :` $\exists c \cdot c \in graph \to C \land c \in valid$

is sufficient for our cause, and the variable `coloring` as well as the assignments to it and the invariants concering it could be removed without loss.

It would have been cleaner to introduce the stages much earlier, possibly after the m01 Colorops refinement. This would allow to define invariants about the variable c and d, the fan or the cd path only when they are actually valid. For example, we do not care about c and d in stage 3, and the properties of the *cd*-path are only relevant in stage 2. In the current way, some invariants about the *cd*-path are predicated by $pathl > 0$, e.g. using the empty path whenever the path variable is actually irrelevant.

Additionally, this would allow to prove termination for the events of each stage independently, therefore the work-arounds of the kind $(min(\{stage, 2\}) - 1)$ would be unnecessary and most variants could be expressed as sets instead of natural numbers, avoiding the poorly supported cardinality operator.

As mentioned above, the introduction of the set *valid* for a valid coloring was less helpful than expected. Especially if the first model would not contain an `coloring` variable, the properties of a valid coloring would only occur in the guard of *finish* and the invariants of the first refinement, which is an acceptable level of repetition.

References

[isa] *Isabelle: A Generic Theorem Proving Environment.* `http://isabelle.in.tum.de/`, . – Version 2009-2

[MG90] Misra, J ; Gries, David: A Constructive Proof of Vizing's Theorem. In: *Information Processing Letters* 41 (1990). `http://www.cs.utexas.edu/users/psp/vizing.pdf`

[RD90] Rao, Josyula R. ; Dijkstra, Edsger W.: *Constructing the proof of Vizing's Theorem.* `http://www.cs.utexas.edu/users/EWD/ewd10xx/EWD1075.PDF`. Version: Februar 1990

A. The models

This appendix contains the details of the models. The comments are part of the model and included here by the LaTeX plugin. Information about the generated proof obligations is not available.

A.1. Input

CONTEXT Input

This context defines the parameters of our algorithm, e.g. the graph for which we have to find a suitable coloring.

SETS

`V` nodes of the graph

`C` available colors

CONSTANTS

`graph` The graph, represented as a relation over the nodes

`N` The maximum degree of the graph

`valid` This is a convenience definition, the set of all valid, possibly incomplete, colorings of the graph. This is required as Rodin does not allow for an easy way to define custom predicates, especially not some that are define with respect to an user-introduced constant.

AXIOMS

`axm1` : $finite(V)$

We only consider finite graphs.

`axm2` : $finite(C)$

And the number of available colors is finite, too (otherwise we may not talk about its cardinality).

`axm3` : $graph \in (V \leftrightarrow V)$

Type specification for the constant graph.

`axm4` : $finite(graph)$

This would follow from the previous two axioms. We include it for convenience.

`axm5` : $graph^{-1} = graph$

Our graph is undirected. We represent that as a directed graph where all inverted edges are present.

`axm6` : $\forall x \cdot card(graph[\{x\}]) \leq N$

The degree of each node is at most N,...

`axm7` : $N \in \mathbb{N}$

...which is a natural number.

`axm8` : $\forall x \cdot (x \mapsto x) \notin graph$

The graph has no autoloops.

`axm9` : $card(C) = N + 1$

The number of available colors is larger than the largest degree.

`axm10` : $valid \in \mathbb{P}(graph \nrightarrow C)$

We introduce a constant to be able to concisely write that a coloring is valid.

axm11 : $valid = \{c|$
$c \in graph \nrightarrow C$
$\land (\forall x \cdot \forall y \cdot \forall z \cdot \forall d \cdot ((x \mapsto y) \mapsto d) \in c \land ((x \mapsto z) \mapsto d) \in c \Rightarrow y = z)$
$\land (\forall x \cdot \forall y \cdot \forall d \cdot ((x \mapsto y) \mapsto d) \in c \Rightarrow ((y \mapsto x) \mapsto d) \in c)\}$

A coloring is valid if: (1) It is a partial function from the graph edges to the set of colors. (2) Different edges on one node have different colors. (3) It is consistent with the fact that we actually consider undirected graphs.

axm12 : $graph \neq \varnothing$

Not part of the official specification, but otherwise we have nothing to prove anyways, and we need this to assign varibles later.

axm13 : $\forall c \cdot c \in valid \Rightarrow (\forall y \cdot \exists d \cdot \forall z \cdot y \mapsto z \mapsto d \notin c)$

This should be provable from the above definitions.

END

A.2. m00 Spec

MACHINE m00_Spec

The first model gives the abstract specification of the problem.

SEES Input

VARIABLES

coloring

INVARIANTS

inv1 : $coloring \in graph \nrightarrow C$

inv2 : $coloring \in valid$

EVENTS

Initialisation

begin

act1 : $coloring := \varnothing$

end

Event *finish* $\widehat{=}$

When we know that there is a valid coloring that colors the whole graph, we are done.

when

grd1 : $\exists c \cdot c \in graph \rightarrow C \land c \in valid$

then

act1 : $coloring :| coloring' \in graph \rightarrow C \land coloring' \in valid$

end

END

A.3. m01 Colorops

MACHINE m01_Colorops

Definition of the operations on the coloring along the way.

REFINES m00_Spec

SEES Input

VARIABLES

coloring

coloring2 This is the mutable coloring that we work on during the execution of the program.

INVARIANTS

inv1 : $coloring2 \in valid$

The coloring will always stay valid.

EVENTS

Initialisation

extended

We start with an empty coloring.

begin

act1 : coloring $:= \varnothing$

act2 : $coloring2 := \varnothing$

end

Event *finish* $\widehat{=}$

And we are done once all edges are colored.

refines *finish*

when

grd1 : $dom(coloring2) = graph$

then

act1 : $coloring := coloring2$

end

Event *color1* $\widehat{=}$

This event colors an edge that can be colored without invalidating the coloring.

Status convergent

any

x

y

d

where

grd1 : $x \mapsto y \in graph$

grd2 : $x \mapsto y \notin dom(coloring2)$

grd3 : $\forall z \cdot (x \mapsto z) \mapsto d \notin coloring2$

grd4 : $\forall z \cdot (y \mapsto z) \mapsto d \notin coloring2$

then

act1 : $coloring2 := coloring2 \cup \{x \mapsto y \mapsto d, y \mapsto x \mapsto d\}$

end

Event *colormove* $\widehat{=}$

An edge may be colored by simultanously uncoloring a neighboring edge, if the color is free on the other side.

Status anticipated

any

x

y

w

where

grd1 : $x \mapsto y \in graph$

grd2 : $x \mapsto w \in graph$

grd3 : $x \mapsto y \notin dom(coloring2)$

grd4 : $x \mapsto w \in dom(coloring2)$

grd5 : $\forall z \cdot \neg(y \mapsto z \mapsto coloring2(x \mapsto w)) \in coloring2$

then

act1 : $coloring2 := (\{x \mapsto w, w \mapsto x\} \lhd coloring2) \cup \{x \mapsto y \mapsto coloring2(x \mapsto w), y \mapsto x \mapsto coloring2(x \mapsto w)\}$

end

Event *invertpath* $\widehat{=}$

Two colors c and d may be flipped in a region of the graph that is closed with regard to those two colors.

Status anticipated

any

c

d

$path$

where

grd1 : $c \in C$

grd2 : $d \in C$

grd4 : $path \subseteq V$

grd3 : $\forall y \cdot y \in path \Rightarrow (\forall z \cdot ((y \mapsto z \mapsto c \in coloring2 \vee y \mapsto z \mapsto d \in coloring2) \Rightarrow z \in path))$

The path is closed with regard to these colors.

then

act1 : $coloring2 := \{(y \mapsto z \mapsto e') | \exists e \cdot (y \mapsto z \mapsto e) \in coloring2 \wedge (((y \in path \vee z \in path) \wedge ((e = d \wedge e' = c) \vee (e = c \wedge e' = d) \vee (e \neq d \wedge e \neq c \wedge e = e'))) \vee (y \notin path \wedge z \notin path \wedge e' = e)) \}$

As one can guess, this definition causes proofs to be come difficult.

end

VARIANT

`card(graph) − card(dom(coloring2))`

We never decrease the number of colored edges, so this terminates eventually.

END

A.4. m02 FixX

MACHINE m02_FixX

Fixes the variable X. Events are refined to remove X from the list of parameters.

REFINES m01_Colorops

SEES Input

VARIABLES

`coloring`

`coloring2`

`X` This vertex is on one side of the uncolored edge and is not altered during any recolorings.

INVARIANTS

inv1 : $X \in V$

X is a vertex.

inv2 : $dom(coloring2) = graph \lor (\exists y \cdot (X \mapsto y \in graph \land X \mapsto y \notin dom(coloring2)))$

Unless we are already done, there is an uncolored edge from this node.

EVENTS

Initialisation

extended

begin

act1 : `coloring` $:= \varnothing$

act2 : `coloring2` $:= \varnothing$

act3 : $X :| \exists y \cdot X' \mapsto y \in graph$

end

Event *finish* $\widehat{=}$

extends *finish*

when

grd1 : `dom(coloring2) = graph`

then

act1 : `coloring := coloring2`

end

Event *color1* $\widehat{=}$

refines *color1*

any

y

d

where

grd1 : $X \mapsto y \in graph$

grd2 : $X \mapsto y \notin dom(coloring2)$

grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$

grd4 : $\forall z \cdot (y \mapsto z) \mapsto d \notin coloring2$

with

x : `x = X`

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto y \mapsto d, y \mapsto X \mapsto d\}$

act2 : $X :| dom(coloring2 \cup \{X \mapsto y \mapsto d, y \mapsto X \mapsto d\}) = graph \lor (\exists z \cdot X' \mapsto z \in graph \setminus dom(coloring2 \cup \{X \mapsto y \mapsto d, y \mapsto X \mapsto d\}))$

After we colored the edge, we need to find a new uncolored X, at least unless the graph is fully colored. At this point it would be nice to be able to refer to coloring2'.

end

Event *colormove* $\widehat{=}$
Status anticipated
refines *colormove*

any

y

w

where

grd1 : $X \mapsto y \in graph$

grd2 : $X \mapsto w \in graph$

grd3 : $X \mapsto y \notin dom(coloring2)$

grd4 : $X \mapsto w \in dom(coloring2)$

grd5 : $\forall z \cdot \neg(y \mapsto z \mapsto coloring2(X \mapsto w)) \in coloring2$

with

x : $\mathtt{x} = \mathtt{X}$

then

act1 : $coloring2 := (\{X \mapsto w, w \mapsto X\} \lhd coloring2) \cup \{X \mapsto y \mapsto coloring2(X \mapsto w), y \mapsto X \mapsto coloring2(X \mapsto w)\}$

end

Event *invertpath* $\widehat{=}$
Status anticipated
extends *invertpath*

any

c

d

path

where

grd1 : $\mathtt{c} \in \mathtt{C}$

grd2 : $\mathtt{d} \in \mathtt{C}$

grd4 : $\mathtt{path} \subseteq \mathtt{V}$

grd3 : $\forall \mathtt{y} \cdot \mathtt{y} \in \mathtt{path} \Rightarrow (\forall \mathtt{z} \cdot ((\mathtt{y} \mapsto \mathtt{z} \mapsto \mathtt{c} \in \mathtt{coloring2} \lor \mathtt{y} \mapsto \mathtt{z} \mapsto \mathtt{d} \in \mathtt{coloring2}) \Rightarrow \mathtt{z} \in \mathtt{path}))$

The path is closed with regard to these colors.

then

act1 : $\mathtt{coloring2} := \{(\mathtt{y} \mapsto \mathtt{z} \mapsto \mathtt{e'}) \mid \exists \mathtt{e} \cdot (\mathtt{y} \mapsto \mathtt{z} \mapsto \mathtt{e}) \in \mathtt{coloring2} \land (((\mathtt{y} \in \mathtt{path} \lor \mathtt{z} \in \mathtt{path}) \land ((\mathtt{e} = \mathtt{d} \land \mathtt{e'} = \mathtt{c}) \lor (\mathtt{e} = \mathtt{c} \land \mathtt{e'} = \mathtt{d}) \lor (\mathtt{e} \neq \mathtt{d} \land \mathtt{e} \neq \mathtt{c} \land \mathtt{e} = \mathtt{e'}))) \lor (\mathtt{y} \notin \mathtt{path} \land \mathtt{z} \notin \mathtt{path} \land \mathtt{e'} = \mathtt{e}))\}$

As one can guess, this definition causes proofs to be come difficult.

end

END

A.5. m03 FixY

MACHINE m03_FixY

Fixes the variable Y (which changes with color1 and colormove), and removes it from

parameter lists.
REFINES m02_FixX
SEES Input
VARIABLES

coloring
coloring2
X
Y

INVARIANTS

inv1 : $dom(coloring2) = graph \lor X \mapsto Y \in graph \setminus dom(coloring2)$

EVENTS
Initialisation

begin

act1 : $coloring := \varnothing$
act2 : $coloring2 := \varnothing$
act3 : $X, Y :| X' \mapsto Y' \in graph$

end

Event *finish* $\widehat{=}$
extends *finish*

when

grd1 : $dom(coloring2) = graph$

then

act1 : $coloring := coloring2$

end

Event *color1* $\widehat{=}$
refines *color1*

any

d

where

grd1 : $X \mapsto Y \in graph$
grd2 : $X \mapsto Y \notin dom(coloring2)$
grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$
grd4 : $\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2$

with

y : $y = Y$

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
act2 : $X, Y :| dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}) = graph \lor X' \mapsto Y' \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})$

end

Event *colormove* $\widehat{=}$
Status anticipated
refines *colormove*

any

w

where

grd1 : $X \mapsto Y \in graph$
grd2 : $X \mapsto w \in graph$
grd3 : $X \mapsto Y \notin dom(coloring2)$
grd4 : $X \mapsto w \in dom(coloring2)$
grd5 : $\forall z \cdot \neg(Y \mapsto z \mapsto coloring2(X \mapsto w)) \in coloring2$

with

y : $y = Y$

then

act1 : $coloring2 := (\{X \mapsto w, w \mapsto X\} \lhd coloring2) \cup \{X \mapsto Y \mapsto coloring2(X \mapsto w), Y \mapsto X \mapsto coloring2(X \mapsto w)\}$
act2 : $Y := w$

end

Event *invertpath* $\widehat{=}$
Status anticipated
extends *invertpath*

any

c
d
path

where

grd1 : $c \in C$
grd2 : $d \in C$
grd4 : $path \subseteq V$
grd3 : $\forall y \cdot y \in path \Rightarrow (\forall z \cdot ((y \mapsto z \mapsto c \in coloring2 \vee y \mapsto z \mapsto d \in coloring2) \Rightarrow z \in path))$
The path is closed with regard to these colors.

then

act1 : $coloring2 := \{(y \mapsto z \mapsto e') \mid \exists e \cdot (y \mapsto z \mapsto e) \in coloring2 \wedge (((y \in path \vee z \in path) \wedge ((e = d \wedge e' = c) \vee (e = c \wedge e' = d) \vee (e \neq d \wedge e \neq c \wedge e = e'))) \vee (y \notin path \wedge z \notin path \wedge e' = e)) \}$
As one can guess, this definition causes proofs to be come difficult.

end

END

A.6. m04 Fan

MACHINE m04_Fan
This machine introduces the concept of the fan.
REFINES m03_FixY
SEES Input
VARIABLES

coloring
coloring2

`X`

`Y`

`fan` The fan.

`l` The size of the fan.

INVARIANTS

inv3 : $X \mapsto Y \notin dom(coloring2) \Rightarrow l \in \mathbb{N}_1$

Some of these things only hold if we are not already done, thus the implication in the invariants.

inv9 : $X \mapsto Y \in dom(coloring2) \Rightarrow l = 0$

If we are done, the fan should be the empty function.

inv2 : $fan \in 1 \,..\, l \rightarrowtail V$

inv1 : $X \mapsto Y \notin dom(coloring2) \Rightarrow (fan(1) = Y)$

The fan always starts with Y.

inv4 : $\forall n \cdot n \in 1 \,..\, l \Rightarrow X \mapsto fan(n) \in graph$

The fan contains neighbours of X.

inv5 : $\forall n \cdot n \in 2 \,..\, l \Rightarrow X \mapsto fan(n) \in dom(coloring2)$

All edges to fan vertices but the first are colored.

inv6 : $\forall n \cdot n \in 1 \,..(l-1) \Rightarrow (\forall z \cdot \forall d \cdot (fan(n) \mapsto z \mapsto d) \in coloring2 \Rightarrow (X \mapsto fan(n+1) \mapsto d) \notin coloring2)$

And the color of such an edge is free on the preceding vertex on the fan.

EVENTS

Initialisation

begin

act1 : $coloring := \varnothing$

act2 : $coloring2 := \varnothing$

act3 : $X, Y, fan : |(X' \mapsto Y' \in graph \land fan' = \{1 \mapsto Y'\})$

act4 : $l := 1$

end

Event *finish* $\widehat{=}$

extends *finish*

when

grd1 : `dom(coloring2) = graph`

then

act1 : `coloring := coloring2`

end

Event *color1a* $\widehat{=}$

At this point, the color1 event is split into one that finishes the algorithm, and one where we have to continue.

refines *color1*

any

d

where

grd1 : $X \mapsto Y \in graph$

grd2 : $X \mapsto Y \notin dom(coloring2)$

grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$

grd4 : $\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2$
grd5 : $dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}) = graph$
This is the last uncolored edge.

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
act4 : $fan := \varnothing$
We remove the fan.
act3 : $l := 0$

end

Event *color1b* $\widehat{=}$
This is not the last edge.
refines *color1*

any

d
$X2$ So here is our next edge to be colored.
$Y2$

where

grd1 : $X \mapsto Y \in graph$
grd2 : $X \mapsto Y \notin dom(coloring2)$
grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$
grd4 : $\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2$
grd5 : $X2 \mapsto Y2 \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})$

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
act2 : $X := X2$
act3 : $Y := Y2$
act4 : $fan := \{1 \mapsto Y2\}$
The fan is initialized with the new value of Y.
act5 : $l := 1$

end

Event *colormove* $\widehat{=}$
This refinement ensures that colormove is only applied to the first edge of the fan, thus removing one paramter.
Status anticipated
refines *colormove*

when

grd1 : $X \mapsto Y \in graph$
grd3 : $X \mapsto Y \notin dom(coloring2)$
grd4 : $l \geq 2$

with

w : $w = fan(2)$

then

act1 : $coloring2 := (\{X \mapsto fan(2), fan(2) \mapsto X\} \lhd coloring2) \cup \{X \mapsto Y \mapsto coloring2(X \mapsto fan(2)), Y \mapsto X \mapsto coloring2(X \mapsto fan(2))\}$
act2 : $Y := fan(2)$
act3 : $fan := (\lambda n \cdot n \in 1 \,..\, l-1 | fan(n+1))$
The fan is updated afterwards.

act4 : $l := l - 1$

end

Event *invertpath_k* $\widehat{=}$

We only invert a cd-path if there is an edge of color d towards the fan (otherwise the cd-path is empty anyways).

Status anticipated

extends *invertpath*

any

c

d

path

k Fan edge with color d

where

grd1 : $c \in C$

grd2 : $d \in C$

grd4 : $path \subseteq V$

grd3 : $\forall y \cdot y \in path \Rightarrow (\forall z \cdot ((y \mapsto z \mapsto c \in coloring2 \lor y \mapsto z \mapsto d \in coloring2) \Rightarrow z \in path))$

The path is closed with regard to these colors.

grd9 : $l \in \mathbb{N}_1$

grd5 : $X \in path$

grd7 : $\forall z \cdot (fan(l) \mapsto z \mapsto d) \notin coloring2$

grd8 : $\forall z \cdot (X \mapsto z \mapsto c) \notin coloring2$

grd11 : $k \in 1 \,..\, l - 1$

grd10 : $(X \mapsto fan(k + 1) \mapsto d) \in coloring2$

then

act1 : $coloring2 := \{(y \mapsto z \mapsto e') | \exists e \cdot (y \mapsto z \mapsto e) \in coloring2 \land (((y \in path \lor z \in path) \land ((e = d \land e' = c) \lor (e = c \land e' = d) \lor (e \neq d \land e \neq c \land e = e'))) \lor (y \notin path \land z \notin path \land e' = e)) \}$

As one can guess, this definition causes proofs to be come difficult.

act2 : $l, fan : |(fan(k) \in path \Rightarrow l' = l \land fan' = fan) \land (fan(k) \notin path \Rightarrow l' = k \land fan' = 1 \,..\, k \lhd fan)$

We either use the old path, or take a prefix thereof.

end

Event *extendfan* $\widehat{=}$

This event builds up the fan by adding a new vertex, if possible.

Status anticipated

any

z

where

grd4 : $l \in \mathbb{N}_1$

grd1 : $z \notin ran(fan)$

grd2 : $X \mapsto z \in dom(coloring2)$

grd3 : $\forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)$

then

act1 : $l := l + 1$

act2 : $fan := fan \cup \{l + 1 \mapsto z\}$

end

Event *fan_done* $\widehat{=}$

This event does nothing in this refinement, but fires when the fan is fully built. This will later be refined with some action.

Status anticipated

when

grd1 : $l \in \mathbb{N}_1$

grd2 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land \forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)))$

then

skip

end

END

A.7. m05 CDpath as path

MACHINE m05_CDpath_as_path

The c-d-path should be a path

REFINES m04_Fan

SEES Input

VARIABLES

coloring

coloring2

X

Y

fan

l length of fan

EVENTS

Initialisation

extended

begin

act1 : $\texttt{coloring} := \varnothing$

act2 : $\texttt{coloring2} := \varnothing$

act3 : $\texttt{X}, \texttt{Y}, \texttt{fan} : |(\texttt{X}' \mapsto \texttt{Y}' \in \texttt{graph} \land \texttt{fan}' = \{1 \mapsto \texttt{Y}'\})$

act4 : $\texttt{l} := 1$

end

Event *finish* $\widehat{=}$

extends *finish*

when

grd1 : $\texttt{dom}(\texttt{coloring2}) = \texttt{graph}$

then

act1 : $\texttt{coloring} := \texttt{coloring2}$

end

Event *color1a* $\widehat{=}$
extends *color1a*

any

d

where

grd1 : $\mathtt{X} \mapsto \mathtt{Y} \in \mathtt{graph}$
grd2 : $\mathtt{X} \mapsto \mathtt{Y} \notin \mathtt{dom}(\mathtt{coloring2})$
grd3 : $\forall \mathtt{z} \cdot (\mathtt{X} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$
grd4 : $\forall \mathtt{z} \cdot (\mathtt{Y} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$
grd5 : $\mathtt{dom}(\mathtt{coloring2} \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{d}, \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{d}\}) = \mathtt{graph}$

then

act1 : $\mathtt{coloring2} := \mathtt{coloring2} \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{d}, \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{d}\}$
act4 : $\mathtt{fan} := \varnothing$
act3 : $\mathtt{l} := 0$

end

Event *color1b* $\widehat{=}$
extends *color1b*

any

d
X2
Y2

where

grd1 : $\mathtt{X} \mapsto \mathtt{Y} \in \mathtt{graph}$
grd2 : $\mathtt{X} \mapsto \mathtt{Y} \notin \mathtt{dom}(\mathtt{coloring2})$
grd3 : $\forall \mathtt{z} \cdot (\mathtt{X} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$
grd4 : $\forall \mathtt{z} \cdot (\mathtt{Y} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$
grd5 : $\mathtt{X2} \mapsto \mathtt{Y2} \in \mathtt{graph} \setminus \mathtt{dom}(\mathtt{coloring2} \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{d}, \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{d}\})$

then

act1 : $\mathtt{coloring2} := \mathtt{coloring2} \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{d}, \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{d}\}$
act2 : $\mathtt{X} := \mathtt{X2}$
act3 : $\mathtt{Y} := \mathtt{Y2}$
act4 : $\mathtt{fan} := \{1 \mapsto \mathtt{Y2}\}$
act5 : $\mathtt{l} := 1$

end

Event *colormove* $\widehat{=}$
Status anticipated
extends *colormove*

when

grd1 : $\mathtt{X} \mapsto \mathtt{Y} \in \mathtt{graph}$
grd3 : $\mathtt{X} \mapsto \mathtt{Y} \notin \mathtt{dom}(\mathtt{coloring2})$
grd4 : $\mathtt{l} \geq 2$

then

act1 : $\mathtt{coloring2} := (\{\mathtt{X} \mapsto \mathtt{fan}(2), \mathtt{fan}(2) \mapsto \mathtt{X}\} \lhd \mathtt{coloring2}) \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{fan}(2)), \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{fan}(2))\}$
act2 : $\mathtt{Y} := \mathtt{fan}(2)$

act3 : $\mathtt{fan} := (\lambda \mathtt{n} \cdot \mathtt{n} \in 1 \,..\, \mathtt{l} - 1 | \mathtt{fan}(\mathtt{n} + 1))$
act4 : $\mathtt{l} := \mathtt{l} - 1$
end

Event *invertpath_k* $\widehat{=}$
Status anticipated
refines *invertpath_k*

any
c
d
cdp
k Fan edge with color d
pl path length

where
grd13 : $pl \in \mathbb{N}$
grd12 : $cdp \in 0 \,..\, pl \rightarrow V$
grd1 : $c \in C$
grd2 : $d \in C$
grd3 : $\forall i \cdot i \in 1 \,..\, pl - 1 \Rightarrow (\forall z \cdot (cdp(i) \mapsto z \mapsto c \in coloring2 \lor cdp(i) \mapsto z \mapsto d \in coloring2) \Rightarrow (z = cdp(i-1) \lor z = cdp(i+1)))$
grd9 : $l \in \mathbb{N}_1$
grd5 : $cdp(0) = X$
grd7 : $\forall z \cdot (fan(l) \mapsto z \mapsto d) \notin coloring2$
grd8 : $\forall z \cdot (X \mapsto z \mapsto c) \notin coloring2$
grd11 : $k \in 1 \,..\, l - 1$
grd10 : $(X \mapsto fan(k+1) \mapsto d) \in coloring2$
grd14 : $\forall z \cdot cdp(0) \mapsto z \mapsto d \in coloring2 \Rightarrow (pl > 0 \land z = cdp(1))$
grd15 : $pl > 0 \Rightarrow (\forall z \cdot (cdp(pl) \mapsto z \mapsto c \in coloring2 \lor cdp(pl) \mapsto z \mapsto d \in coloring2) \Rightarrow (z = cdp(pl - 1)))$

with
path : $\mathtt{path} = \mathtt{ran}(\mathtt{cdp})$

then
act1 : $coloring2 := \{(y \mapsto z \mapsto e') | \exists e \cdot (y \mapsto z \mapsto e) \in coloring2 \land (((y \in ran(cdp) \lor z \in ran(cdp)) \land ((e = d \land e' = c) \lor (e = c \land e' = d) \lor (e \neq d \land e \neq c \land e = e'))) \lor (y \notin ran(cdp) \land z \notin ran(cdp) \land e' = e))\}$
act2 : $l, fan :| (fan(k) \in ran(cdp) \Rightarrow l' = l \land fan' = fan) \land (fan(k) \notin ran(cdp) \Rightarrow l' = k \land fan' = 1 \,..\, k \lhd fan)$
end

Event *extendfan* $\widehat{=}$
Status anticipated
extends *extendfan*

any
z

where
grd4 : $\mathtt{l} \in \mathbb{N}_1$
grd1 : $\mathtt{z} \notin \mathtt{ran}(\mathtt{fan})$
grd2 : $\mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom}(\mathtt{coloring2})$
grd3 : $\forall \mathtt{w} \cdot \forall \mathtt{d} \cdot \mathtt{fan}(\mathtt{l}) \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{z})$

then

act1 : $l := l + 1$

act2 : $fan := fan \cup \{l + 1 \mapsto z\}$

end

Event *fan_done* $\widehat{=}$
Status anticipated
extends *fan_done*

when

grd1 : $l \in \mathbb{N}_1$

grd2 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land \forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)))$

then

skip

end

END

A.8. m06 CDpath building

MACHINE m06_CDpath_building
The cd-path is being built up, similar to the fan before.
REFINES m05_CDpath_as_path
SEES Input
VARIABLES

coloring

coloring2

X

Y

fan

l length of fan

cdpath The cd-path (or a prefix of it, while it is being constructed).

pathl The length of the cd-path.

d_ This color is free on the last vertex of the fan.

c_ This color is free on X. Together they define the cd-path.

INVARIANTS

inv6 : $d\_ \in C$

inv5 : $c\_ \in C$

inv1 : $cdpath \in 0 .. pathl \rightarrowtail V$

inv2 : $pathl \in \mathbb{N}$

inv3 : $cdpath(0) = X$
The cd-path always starts with X.

inv4 : $\forall i \cdot i \in 1 .. pathl - 1 \Rightarrow (\forall z \cdot (cdpath(i) \mapsto z \mapsto c\_ \in coloring2 \lor cdpath(i) \mapsto z \mapsto d\_ \in coloring2) \Rightarrow (z = cdpath(i - 1) \lor z = cdpath(i + 1)))$
An edge with color c or d causes that edge to enlargen the cdpath.

inv7 : $pathl > 0 \Rightarrow (\forall z \cdot cdpath(0) \mapsto z \mapsto d\_ \in coloring2 \Rightarrow z = cdpath(1))$
From X, the cd-path extends along edges of color d, as c is free on X.

inv8 : $pathl > 0 \Rightarrow (\forall z \cdot (X \mapsto z \mapsto c\_) \notin coloring2)$
c is free on X, at least once we started to build up the cd-path.

inv9 : $\forall i \cdot i \in 1 .. pathl \Rightarrow (cdpath(i) \mapsto cdpath(i-1) \mapsto c\_ \in coloring2 \lor cdpath(i) \mapsto cdpath(i-1) \mapsto d\_ \in coloring2)$
Vertices on the cd-path are connected by an edge of color c or d.

inv10 : $X \mapsto Y \notin dom(coloring2) \Rightarrow (\forall z \cdot (fan(l) \mapsto z \mapsto d\_) \notin coloring2)$
Unless we are done, d is free on the last vertex of the fan.

EVENTS
Initialisation

begin
- act1 : $coloring := \varnothing$
- act2 : $coloring2 := \varnothing$
- act3 : $X, Y, fan, cdpath : | X' \mapsto Y' \in graph \land fan' = \{1 \mapsto Y'\} \land cdpath' = \{0 \mapsto X'\}$
- act4 : $l := 1$
- act5 : $pathl := 0$
- act6 : $c\_ :\in C$
- act7 : $d\_ :\in C$

end

Event *finish* $\widehat{=}$
extends *finish*

when
- grd1 : $\mathtt{dom(coloring2) = graph}$

then
- act1 : $\mathtt{coloring := coloring2}$

end

Event *color1a* $\widehat{=}$
extends *color1a*

any
d

where
- grd1 : $\mathtt{X \mapsto Y \in graph}$
- grd2 : $\mathtt{X \mapsto Y \notin dom(coloring2)}$
- grd3 : $\mathtt{\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2}$
- grd4 : $\mathtt{\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2}$
- grd5 : $\mathtt{dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}) = graph}$
  This is the last uncolored edge.
- grd6 : $pathl = 0$

then
- act1 : $\mathtt{coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}}$
- act4 : $\mathtt{fan := \varnothing}$
  We remove the fan.
- act3 : $\mathtt{l := 0}$

end

Event *color1b* $\widehat{=}$
extends *color1b*

any

d
X2 So here is our next edge to be colored.
Y2

where

grd1 : $X \mapsto Y \in graph$
grd2 : $X \mapsto Y \notin dom(coloring2)$
grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$
grd4 : $\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2$
grd5 : $X2 \mapsto Y2 \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})$

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
act2 : $X := X2$
act3 : $Y := Y2$
act4 : $fan := \{1 \mapsto Y2\}$
The fan is initialized with the new value of Y.
act5 : $l := 1$
act6 : $cdpath := \{0 \mapsto X2\}$
act7 : $pathl := 0$
act8 : $d\_ :| \forall z \cdot (Y2 \mapsto z \mapsto d\_') \notin coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
We need to find a new d.

end

Event *colormove* $\widehat{=}$
Status anticipated
extends *colormove*

when

grd1 : $X \mapsto Y \in graph$
grd3 : $X \mapsto Y \notin dom(coloring2)$
grd4 : $l \geq 2$
grd5 : $pathl = 0$

then

act1 : $coloring2 := (\{X \mapsto fan(2), fan(2) \mapsto X\} \lhd coloring2) \cup \{X \mapsto Y \mapsto coloring2(X \mapsto fan(2)), Y \mapsto X \mapsto coloring2(X \mapsto fan(2))\}$
act2 : $Y := fan(2)$
act3 : $fan := (\lambda n \cdot n \in 1 .. l - 1 | fan(n + 1))$
The fan is updated afterwards.
act4 : $l := l - 1$

end

Event *invertpath_k* $\widehat{=}$
We want to use the calculated cd path, of course, so the parameters are replaced.
Status anticipated
refines *invertpath_k*

any

k Fan edge with color d

where

grd16 : $pathl > 0$
grd9 : $l \in \mathbb{N}_1$
grd11 : $k \in 1 \,..\, l - 1$
grd10 : $(X \mapsto fan(k+1) \mapsto d\_) \in coloring2$
grd15 : $\forall z \cdot (cdpath(pathl) \mapsto z \mapsto c\_ \in coloring2 \lor cdpath(pathl) \mapsto z \mapsto d\_ \in coloring2) \Rightarrow (z = cdpath(pathl - 1))$
This ensures that the cd-path is already fully calculated.

with

d : d = d_
c : c = c_
cdp : cdp = cdpath
pl : pl = pathl

then

act1 : $coloring2 := \{(y \mapsto z \mapsto e') | \exists e \cdot (y \mapsto z \mapsto e) \in coloring2 \land (((y \in ran(cdpath) \lor z \in ran(cdpath)) \land ((e = d\_ \land e' = c\_) \lor (e = c\_ \land e' = d\_) \lor (e \neq d\_ \land e \neq c\_ \land e = e'))) \lor (y \notin ran(cdpath) \land z \notin ran(cdpath) \land e' = e))\}$
act2 : $l, fan : |(fan(k) \in ran(cdpath) \Rightarrow l' = l \land fan' = fan) \land (fan(k) \notin ran(cdpath) \Rightarrow l' = k \land fan' = 1 \,..\, k \lhd fan)$
act3 : $cdpath := \{0 \mapsto X\}$
act4 : $pathl := 0$
act5 : $c\_ := d\_$

end

Event *no_invertpath* $\widehat{=}$
Status anticipated

when

grd2 : $l \in \mathbb{N}_1$
grd4 : $\forall z \cdot X \mapsto z \mapsto d\_ \notin coloring2$
d is free on X, so no path to build.
grd5 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land (\forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z))))$
This ensures that the fan is maximal.

then

act4 : $c\_ := d\_$
If the fan is maximal and d free on X, then the cd-path is empty and we conclude that d is free on X.

end

Event *extendfan* $\widehat{=}$
Status anticipated
extends *extendfan*

any

z

where

grd4 : $l \in \mathbb{N}_1$
grd1 : $z \notin ran(fan)$
grd2 : $X \mapsto z \in dom(coloring2)$
grd3 : $\forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)$
grd5 : $pathl = 0$

then

act1 : $l := l + 1$

act2 : $fan := fan \cup \{l + 1 \mapsto z\}$

act3 : $d\_ :| \forall w \cdot z \mapsto w \mapsto d\_' \notin coloring2$

We need to update d, the free color of the last node on the fan.

end

Event *extendcdpath* $\widehat{=}$

This builds the cd-path by extending it along edges of color c or d.

Status anticipated

any

z

where

grd1 : $z \in V$

grd2 : $z \notin ran(cdpath)$

grd3 : $cdpath(pathl) \mapsto z \mapsto c\_ \in coloring2 \lor cdpath(pathl) \mapsto z \mapsto d\_ \in coloring2$

grd4 : $\forall z \cdot X \mapsto z \mapsto c\_ \notin coloring2$

then

act1 : $cdpath := cdpath \cup \{pathl + 1 \mapsto z\}$

act2 : $pathl := pathl + 1$

end

Event *fan_done* $\widehat{=}$

Status anticipated

extends *fan_done*

when

grd1 : $l \in \mathbb{N}_1$

grd2 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land \forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)))$

grd3 : $pathl = 0$

then

act1 : $c\_ :| \forall z \cdot X \mapsto z \mapsto c\_' \notin coloring2$

end

END

A.9. m07 Event ordering

MACHINE m07_Event_ordering

This refinement tightens the guards to run the events in the desired order.

REFINES m06_CDpath_building

SEES Input

VARIABLES

coloring

coloring2

X

Y

fan

`l` length of fan

`cdpath` cd-path (or prefix while building) (I am running ot of names)

`pathl` length of cd path

`d_`

`c_`

`stage` stage 3: buiding fan, stage 2: building cd-path and inverting it, stage 1: moving color along path

INVARIANTS

inv1 : $stage \in 1 \,..\, 3$

inv4 : $stage = 3 \Rightarrow pathl = 0$

In strage three, the cd-path stays empty.

inv2 : $stage = 2 \Rightarrow l \in \mathbb{N}_1 \wedge \neg(\exists z \cdot (z \notin ran(fan) \wedge X \mapsto z \in dom(coloring2) \wedge \forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)))$

In stage 2, the fan is fully calculated.

inv5 : $stage = 2 \Rightarrow (\forall z \cdot X \mapsto z \mapsto c\_ \notin coloring2)$

And c is free on X.

inv3 : $stage = 1 \Rightarrow c\_ = d\_$

In stage one, we have d free on both X and the end of the fan.

inv6 : $stage = 1 \Rightarrow pathl = 0$

And in stage 1, the cd-path has been flipped or was empty in the first place.

inv7 : $stage = 1 \Rightarrow (\forall z \cdot X \mapsto z \mapsto c\_ \notin coloring2)$

EVENTS

Initialisation

extended

begin

act1 : $\mathtt{coloring} := \varnothing$

act2 : $\mathtt{coloring2} := \varnothing$

act3 : $\mathtt{X, Y, fan, cdpath} : |\mathtt{X}' \mapsto \mathtt{Y}' \in \mathtt{graph} \wedge \mathtt{fan}' = \{1 \mapsto \mathtt{Y}'\} \wedge \mathtt{cdpath}' = \{0 \mapsto \mathtt{X}'\}$

act4 : $\mathtt{l} := 1$

act5 : $\mathtt{pathl} := 0$

act6 : $\mathtt{c\_} :\in \mathtt{C}$

act7 : $\mathtt{d\_} :\in \mathtt{C}$

act8 : $stage := 3$

end

Event *finish* $\widehat{=}$

extends *finish*

when

grd1 : $\mathtt{dom(coloring2)} = \mathtt{graph}$

then

act1 : $\mathtt{coloring} := \mathtt{coloring2}$

end

Event *color1a* $\widehat{=}$

extends *color1a*

any

d

where

grd1 : $X \mapsto Y \in graph$

grd2 : $X \mapsto Y \notin dom(coloring2)$

grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$

grd4 : $\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2$

grd5 : $dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}) = graph$

This is the last uncolored edge.

grd6 : $pathl = 0$

grd7 : $stage = 1$

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$

act4 : $fan := \varnothing$

We remove the fan.

act3 : $l := 0$

act5 : $stage := 3$

end

Event *color1b* $\widehat{=}$
extends *color1b*

any

d

X2 So here is our next edge to be colored.

Y2

where

grd1 : $X \mapsto Y \in graph$

grd2 : $X \mapsto Y \notin dom(coloring2)$

grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$

grd4 : $\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2$

grd5 : $X2 \mapsto Y2 \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})$

grd6 : $stage = 1$

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$

act2 : $X := X2$

act3 : $Y := Y2$

act4 : $fan := \{1 \mapsto Y2\}$

The fan is initialized with the new value of Y.

act5 : $l := 1$

act6 : $cdpath := \{0 \mapsto X2\}$

act7 : $pathl := 0$

act8 : $d\_ :| \forall z \cdot (Y2 \mapsto z \mapsto d\_') \notin coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$

We need to find a new d.

act9 : $stage := 3$

end

Event *colormove* $\widehat{=}$
Status anticipated
extends *colormove*

when

grd1 : $X \mapsto Y \in graph$
grd3 : $X \mapsto Y \notin dom(coloring2)$
grd4 : $l \geq 2$
grd5 : $pathl = 0$
grd6 : $stage = 1$

then

act1 : $coloring2 := (\{X \mapsto fan(2), fan(2) \mapsto X\} \lhd coloring2) \cup \{X \mapsto Y \mapsto coloring2(X \mapsto fan(2)), Y \mapsto X \mapsto coloring2(X \mapsto fan(2))\}$
act2 : $Y := fan(2)$
act3 : $fan := (\lambda n \cdot n \in 1 .. l - 1 | fan(n + 1))$
The fan is updated afterwards.
act4 : $l := l - 1$

end

Event *invertpath_k* $\widehat{=}$
Status anticipated
extends *invertpath_k*

any

k Fan edge with color d

where

grd16 : $pathl > 0$
grd9 : $l \in \mathbb{N}_1$
grd11 : $k \in 1 .. l - 1$
grd10 : $(X \mapsto fan(k + 1) \mapsto d\_) \in coloring2$
grd15 : $\forall z \cdot (cdpath(pathl) \mapsto z \mapsto c\_ \in coloring2 \lor cdpath(pathl) \mapsto z \mapsto d\_ \in coloring2) \Rightarrow (z = cdpath(pathl - 1))$
This ensures that the cd-path is already fully calculated.
grd17 : $stage = 2$

then

act1 : $coloring2 := \{(y \mapsto z \mapsto e') | \exists e \cdot (y \mapsto z \mapsto e) \in coloring2 \land (((y \in ran(cdpath) \lor z \in ran(cdpath)) \land ((e = d\_ \land e' = c\_) \lor (e = c\_ \land e' = d\_) \lor (e \neq d\_ \land e \neq c\_ \land e = e'))) \lor (y \notin ran(cdpath) \land z \notin ran(cdpath) \land e' = e))\}$
act2 : $l, fan :| (fan(k) \in ran(cdpath) \Rightarrow l' = l \land fan' = fan) \land (fan(k) \notin ran(cdpath) \Rightarrow l' = k \land fan' = 1 .. k \lhd fan)$
act3 : $cdpath := \{0 \mapsto X\}$
act4 : $pathl := 0$
act5 : $c\_ := d\_$
act6 : $stage := 1$

end

Event *extendfan* $\widehat{=}$
Fan is extended first, as long as d_ is not c_ and it can be extended
Status anticipated
extends *extendfan*

any

z

where

grd4 : $l \in \mathbb{N}_1$
grd1 : $z \notin ran(fan)$

grd2 : $X \mapsto z \in dom(coloring2)$
grd3 : $\forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)$
grd5 : $pathl = 0$
grd7 : $stage = 3$

then

act1 : $l := l + 1$
act2 : $fan := fan \cup \{l + 1 \mapsto z\}$
act3 : $d\_ :| \forall w \cdot z \mapsto w \mapsto d\_' \notin coloring2$
We need to update d, the free color of the last node on the fan.

end

Event *extendcdpath* $\widehat{=}$
cd-path is calculated if fan is maximal and c_ is not d_
Status anticipated
extends *extendcdpath*

any
z
where

grd1 : $z \in V$
grd2 : $z \notin ran(cdpath)$
grd3 : $cdpath(pathl) \mapsto z \mapsto c\_ \in coloring2 \lor cdpath(pathl) \mapsto z \mapsto d\_ \in coloring2$
grd4 : $\forall z \cdot X \mapsto z \mapsto c\_ \notin coloring2$
grd7 : $stage = 2$
grd5 : $c\_ \neq d\_$

then

act1 : $cdpath := cdpath \cup \{pathl + 1 \mapsto z\}$
act2 : $pathl := pathl + 1$

end

Event *fan_done* $\widehat{=}$
Status anticipated
extends *fan_done*

when

grd1 : $l \in \mathbb{N}_1$
grd2 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land \forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)))$
grd3 : $pathl = 0$
grd4 : $stage = 3$

then

act1 : $c\_ :| \forall z \cdot X \mapsto z \mapsto c\_' \notin coloring2$
act2 : $stage := 2$

end

Event *noinvertpath* $\widehat{=}$
Status anticipated
extends *no_invertpath*

when

grd2 : $l \in \mathbb{N}_1$

grd4 : $\forall z \cdot X \mapsto z \mapsto d\_ \notin coloring2$
d is free on X, so no path to build.
grd5 : $\neg(\exists z \cdot (z \notin ran(fan) \wedge X \mapsto z \in dom(coloring2) \wedge (\forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z))))$
This ensures that the fan is maximal.
grd1 : *stage = 2*

then

act4 : $c\_ := d\_$
If the fan is maximal and d free on X, then the cd-path is empty and we conclude that d is free on X.
act1 : *stage := 1*

end

END

A.10. m08 Conv Fan Build

MACHINE m08_Conv_Fan_Build
This refinement shows the convergence of fan building.
REFINES m07_Event_ordering
SEES Input
VARIABLES

coloring
coloring2
X
Y
fan
l length of fan
cdpath cd-path (or prefix while building) (I am running ot of names)
pathl length of cd path
d_
c_
stage stage 3: buiding fan, stage 2: building cd-path and inverting it, stage 1: moving color along path

EVENTS
Initialisation

extended

begin

act1 : $coloring := \varnothing$
act2 : $coloring2 := \varnothing$
act3 : $X, Y, fan, cdpath : |X' \mapsto Y' \in graph \wedge fan' = \{1 \mapsto Y'\} \wedge cdpath' = \{0 \mapsto X'\}$
act4 : $l := 1$
act5 : $pathl := 0$
act6 : $c\_ :\in C$
act7 : $d\_ :\in C$
act8 : $stage := 3$

end

Event *finish* $\widehat{=}$
extends *finish*

when

grd1 : $\mathtt{dom(coloring2) = graph}$

then

act1 : $\mathtt{coloring := coloring2}$

end

Event *color1a* $\widehat{=}$
extends *color1a*

any

d

where

grd1 : $\mathtt{X \mapsto Y \in graph}$

grd2 : $\mathtt{X \mapsto Y \notin dom(coloring2)}$

grd3 : $\mathtt{\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2}$

grd4 : $\mathtt{\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2}$

grd5 : $\mathtt{dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}) = graph}$

This is the last uncolored edge.

grd6 : $\mathtt{pathl = 0}$

grd7 : $\mathtt{stage = 1}$

then

act1 : $\mathtt{coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}}$

act4 : $\mathtt{fan := \varnothing}$

We remove the fan.

act3 : $\mathtt{l := 0}$

act5 : $\mathtt{stage := 3}$

end

Event *color1b* $\widehat{=}$
extends *color1b*

any

d

X2 So here is our next edge to be colored.

Y2

where

grd1 : $\mathtt{X \mapsto Y \in graph}$

grd2 : $\mathtt{X \mapsto Y \notin dom(coloring2)}$

grd3 : $\mathtt{\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2}$

grd4 : $\mathtt{\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2}$

grd5 : $\mathtt{X2 \mapsto Y2 \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})}$

grd6 : $\mathtt{stage = 1}$

then

act1 : $\mathtt{coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}}$

act2 : $\mathtt{X := X2}$

act3 : $\mathtt{Y := Y2}$

act4 : $\mathtt{fan := \{1 \mapsto Y2\}}$

The fan is initialized with the new value of Y.

act5 : $l := 1$
act6 : $cdpath := \{0 \mapsto X2\}$
act7 : $pathl := 0$
act8 : $d\_ : |\forall z \cdot (Y2 \mapsto z \mapsto d\_') \notin coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
We need to find a new d.
act9 : $stage := 3$
end

Event *colormove* $\widehat{=}$
Status anticipated
extends *colormove*

when
grd1 : $X \mapsto Y \in graph$
grd3 : $X \mapsto Y \notin dom(coloring2)$
grd4 : $l \geq 2$
grd5 : $pathl = 0$
grd6 : $stage = 1$
then
act1 : $coloring2 := (\{X \mapsto fan(2), fan(2) \mapsto X\} \lhd coloring2) \cup \{X \mapsto Y \mapsto coloring2(X \mapsto fan(2)), Y \mapsto X \mapsto coloring2(X \mapsto fan(2))\}$
act2 : $Y := fan(2)$
act3 : $fan := (\lambda n \cdot n \in 1 \,..\, l-1 | fan(n+1))$
The fan is updated afterwards.
act4 : $l := l - 1$
end

Event *invertpath_k* $\widehat{=}$
Status anticipated
extends *invertpath_k*

any
k Fan edge with color d
where
grd16 : $pathl > 0$
grd9 : $l \in \mathbb{N}_1$
grd11 : $k \in 1 \,..\, l - 1$
grd10 : $(X \mapsto fan(k+1) \mapsto d\_) \in coloring2$
grd15 : $\forall z \cdot (cdpath(pathl) \mapsto z \mapsto c\_ \in coloring2 \lor cdpath(pathl) \mapsto z \mapsto d\_ \in coloring2) \Rightarrow (z = cdpath(pathl - 1))$
This ensures that the cd-path is already fully calculated.
grd17 : $stage = 2$
then
act1 : $coloring2 := \{(y \mapsto z \mapsto e') | \exists e \cdot (y \mapsto z \mapsto e) \in coloring2 \land (((y \in ran(cdpath) \lor z \in ran(cdpath)) \land ((e = d\_ \land e' = c\_) \lor (e = c\_ \land e' = d\_) \lor (e \neq d\_ \land e \neq c\_ \land e = e'))) \lor (y \notin ran(cdpath) \land z \notin ran(cdpath) \land e' = e))\}$
act2 : $l, fan : |(fan(k) \in ran(cdpath) \Rightarrow l' = l \land fan' = fan) \land (fan(k) \notin ran(cdpath) \Rightarrow l' = k \land fan' = 1 \,..\, k \lhd fan)$
act3 : $cdpath := \{0 \mapsto X\}$
act4 : $pathl := 0$
act5 : $c\_ := d\_$

act6 : $\mathtt{stage} := 1$

end

Event *extendfan* $\widehat{=}$

Fan is extended first, as long as d_ is not c_ and it can be extended

Status convergent

extends *extendfan*

any

z

where

grd4 : $\mathtt{l} \in \mathbb{N}_1$

grd1 : $\mathtt{z} \notin \mathtt{ran(fan)}$

grd2 : $\mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom(coloring2)}$

grd3 : $\forall \mathtt{w} \cdot \forall \mathtt{d} \cdot \mathtt{fan(l)} \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2(X} \mapsto \mathtt{z)}$

grd5 : $\mathtt{pathl} = 0$

grd7 : $\mathtt{stage} = 3$

then

act1 : $\mathtt{l} := \mathtt{l} + 1$

act2 : $\mathtt{fan} := \mathtt{fan} \cup \{\mathtt{l} + 1 \mapsto \mathtt{z}\}$

act3 : $\mathtt{d\_} :| \forall \mathtt{w} \cdot \mathtt{z} \mapsto \mathtt{w} \mapsto \mathtt{d\_}' \notin \mathtt{coloring2}$

We need to update d, the free color of the last node on the fan.

end

Event *extendcdpath* $\widehat{=}$

cd-path is calculated if fan is maximal and c_ is not d_

Status anticipated

extends *extendcdpath*

any

z

where

grd1 : $\mathtt{z} \in \mathtt{V}$

grd2 : $\mathtt{z} \notin \mathtt{ran(cdpath)}$

grd3 : $\mathtt{cdpath(pathl)} \mapsto \mathtt{z} \mapsto \mathtt{c\_} \in \mathtt{coloring2} \lor \mathtt{cdpath(pathl)} \mapsto \mathtt{z} \mapsto \mathtt{d\_} \in \mathtt{coloring2}$

grd4 : $\forall \mathtt{z} \cdot \mathtt{X} \mapsto \mathtt{z} \mapsto \mathtt{c\_} \notin \mathtt{coloring2}$

grd7 : $\mathtt{stage} = 2$

grd5 : $\mathtt{c\_} \neq \mathtt{d\_}$

then

act1 : $\mathtt{cdpath} := \mathtt{cdpath} \cup \{\mathtt{pathl} + 1 \mapsto \mathtt{z}\}$

act2 : $\mathtt{pathl} := \mathtt{pathl} + 1$

end

Event *fan_done* $\widehat{=}$

Status anticipated

extends *fan_done*

when

grd1 : $\mathtt{l} \in \mathbb{N}_1$

grd2 : $\neg(\exists \mathtt{z} \cdot (\mathtt{z} \notin \mathtt{ran(fan)} \land \mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom(coloring2)} \land \forall \mathtt{w} \cdot \forall \mathtt{d} \cdot \mathtt{fan(l)} \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2(X} \mapsto \mathtt{z})))$

grd3 : $\mathtt{pathl} = 0$

grd4 : $\texttt{stage} = 3$

then

act1 : $\texttt{c\_} :| \forall \texttt{z} \cdot \texttt{X} \mapsto \texttt{z} \mapsto \texttt{c\_}' \notin \texttt{coloring2}$

act2 : $\texttt{stage} := 2$

end

Event *noinvertpath* $\widehat{=}$
Status anticipated
extends *noinvertpath*

when

grd2 : $\texttt{l} \in \mathbb{N}_1$

grd4 : $\forall \texttt{z} \cdot \texttt{X} \mapsto \texttt{z} \mapsto \texttt{d\_} \notin \texttt{coloring2}$

d is free on X, so no path to build.

grd5 : $\neg(\exists \texttt{z} \cdot (\texttt{z} \notin \texttt{ran(fan)} \land \texttt{X} \mapsto \texttt{z} \in \texttt{dom(coloring2)} \land (\forall \texttt{w} \cdot \forall \texttt{d} \cdot \texttt{fan(l)} \mapsto \texttt{w} \mapsto \texttt{d} \in \texttt{coloring2} \Rightarrow \texttt{d} \neq \texttt{coloring2(X} \mapsto \texttt{z)})))$

This ensures that the fan is maximal.

grd1 : $\texttt{stage} = 2$

then

act4 : $\texttt{c\_} := \texttt{d\_}$

If the fan is maximal and d free on X, then the cd-path is empty and we conclude that d is free on X.

act1 : $\texttt{stage} := 1$

end

VARIANT

$\texttt{max}(\{\texttt{stage} - 2, 0\}) * (\texttt{card}(\texttt{V} \setminus \texttt{ran(fan)}))$

The variant is zero unless in stage 3. then it decreases while the fan size increases.

END

A.11. m09 Conv CDpath

MACHINE m09_Conv_CDpath

This refinement show the convergence of cd-path-building.

REFINES m08_Conv_Fan_Build
SEES Input
VARIABLES

coloring

coloring2

X

Y

fan

l length of fan

cdpath cd-path (or prefix while building) (I am running ot of names)

pathl length of cd path

d_

c_

`stage` stage 3: buiding fan, stage 2: building cd-path and inverting it, stage 1: moving color along path

EVENTS
Initialisation

extended

begin

act1 : $\texttt{coloring} := \varnothing$

act2 : $\texttt{coloring2} := \varnothing$

act3 : $\texttt{X}, \texttt{Y}, \texttt{fan}, \texttt{cdpath} : |\texttt{X}' \mapsto \texttt{Y}' \in \texttt{graph} \wedge \texttt{fan}' = \{1 \mapsto \texttt{Y}'\} \wedge \texttt{cdpath}' = \{0 \mapsto \texttt{X}'\}$

act4 : $\texttt{l} := 1$

act5 : $\texttt{pathl} := 0$

act6 : $\texttt{c\_} :\in \texttt{C}$

act7 : $\texttt{d\_} :\in \texttt{C}$

act8 : $\texttt{stage} := 3$

end

Event *finish* $\widehat{=}$
extends *finish*

when

grd1 : $\texttt{dom}(\texttt{coloring2}) = \texttt{graph}$

then

act1 : $\texttt{coloring} := \texttt{coloring2}$

end

Event *color1a* $\widehat{=}$
extends *color1a*

any

d

where

grd1 : $\texttt{X} \mapsto \texttt{Y} \in \texttt{graph}$

grd2 : $\texttt{X} \mapsto \texttt{Y} \notin \texttt{dom}(\texttt{coloring2})$

grd3 : $\forall \texttt{z} \cdot (\texttt{X} \mapsto \texttt{z}) \mapsto \texttt{d} \notin \texttt{coloring2}$

grd4 : $\forall \texttt{z} \cdot (\texttt{Y} \mapsto \texttt{z}) \mapsto \texttt{d} \notin \texttt{coloring2}$

grd5 : $\texttt{dom}(\texttt{coloring2} \cup \{\texttt{X} \mapsto \texttt{Y} \mapsto \texttt{d}, \texttt{Y} \mapsto \texttt{X} \mapsto \texttt{d}\}) = \texttt{graph}$

This is the last uncolored edge.

grd6 : $\texttt{pathl} = 0$

grd7 : $\texttt{stage} = 1$

then

act1 : $\texttt{coloring2} := \texttt{coloring2} \cup \{\texttt{X} \mapsto \texttt{Y} \mapsto \texttt{d}, \texttt{Y} \mapsto \texttt{X} \mapsto \texttt{d}\}$

act4 : $\texttt{fan} := \varnothing$

We remove the fan.

act3 : $\texttt{l} := 0$

act5 : $\texttt{stage} := 3$

end

Event *color1b* $\widehat{=}$
extends *color1b*

any

d

X2 So here is our next edge to be colored.
Y2

where

grd1 : $X \mapsto Y \in graph$
grd2 : $X \mapsto Y \notin dom(coloring2)$
grd3 : $\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2$
grd4 : $\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2$
grd5 : $X2 \mapsto Y2 \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})$
grd6 : $stage = 1$

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
act2 : $X := X2$
act3 : $Y := Y2$
act4 : $fan := \{1 \mapsto Y2\}$
The fan is initialized with the new value of Y.
act5 : $l := 1$
act6 : $cdpath := \{0 \mapsto X2\}$
act7 : $pathl := 0$
act8 : $d\_ : | \forall z \cdot (Y2 \mapsto z \mapsto d\_') \notin coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
We need to find a new d.
act9 : $stage := 3$

end

Event *colormove* $\widehat{=}$
Status anticipated
extends *colormove*

when

grd1 : $X \mapsto Y \in graph$
grd3 : $X \mapsto Y \notin dom(coloring2)$
grd4 : $l \geq 2$
grd5 : $pathl = 0$
grd6 : $stage = 1$

then

act1 : $coloring2 := (\{X \mapsto fan(2), fan(2) \mapsto X\} \lhd coloring2) \cup \{X \mapsto Y \mapsto coloring2(X \mapsto fan(2)), Y \mapsto X \mapsto coloring2(X \mapsto fan(2))\}$
act2 : $Y := fan(2)$
act3 : $fan := (\lambda n \cdot n \in 1 \,..\, l - 1 | fan(n + 1))$
The fan is updated afterwards.
act4 : $l := l - 1$

end

Event *invertpath_k* $\widehat{=}$
Status anticipated
extends *invertpath_k*

any

k Fan edge with color d

where

grd16 : $pathl > 0$
grd9 : $l \in \mathbb{N}_1$

$\texttt{grd11}: \texttt{k} \in 1 \,..\, \texttt{l} - 1$
$\texttt{grd10}: (\texttt{X} \mapsto \texttt{fan}(\texttt{k}+1) \mapsto \texttt{d\_}) \in \texttt{coloring2}$
$\texttt{grd15}: \forall \texttt{z} \cdot (\texttt{cdpath}(\texttt{pathl}) \mapsto \texttt{z} \mapsto \texttt{c\_} \in \texttt{coloring2} \lor \texttt{cdpath}(\texttt{pathl}) \mapsto \texttt{z} \mapsto \texttt{d\_} \in \texttt{coloring2}) \Rightarrow (\texttt{z} = \texttt{cdpath}(\texttt{pathl} - 1))$
This ensures that the cd-path is already fully calculated.
$\texttt{grd17}: \texttt{stage} = 2$

then

$\texttt{act1}: \texttt{coloring2} := \{(\texttt{y} \mapsto \texttt{z} \mapsto \texttt{e}') | \exists \texttt{e} \cdot (\texttt{y} \mapsto \texttt{z} \mapsto \texttt{e}) \in \texttt{coloring2} \land (((\texttt{y} \in \texttt{ran}(\texttt{cdpath}) \lor \texttt{z} \in \texttt{ran}(\texttt{cdpath})) \land ((\texttt{e} = \texttt{d\_} \land \texttt{e}' = \texttt{c\_}) \lor (\texttt{e} = \texttt{c\_} \land \texttt{e}' = \texttt{d\_}) \lor (\texttt{e} \neq \texttt{d\_} \land \texttt{e} \neq \texttt{c\_} \land \texttt{e} = \texttt{e}'))) \lor (\texttt{y} \notin \texttt{ran}(\texttt{cdpath}) \land \texttt{z} \notin \texttt{ran}(\texttt{cdpath}) \land \texttt{e}' = \texttt{e}))\}$
$\texttt{act2}: \texttt{l}, \texttt{fan} : |(\texttt{fan}(\texttt{k}) \in \texttt{ran}(\texttt{cdpath}) \Rightarrow \texttt{l}' = \texttt{l} \land \texttt{fan}' = \texttt{fan}) \land (\texttt{fan}(\texttt{k}) \notin \texttt{ran}(\texttt{cdpath}) \Rightarrow \texttt{l}' = \texttt{k} \land \texttt{fan}' = 1 \,..\, \texttt{k} \lhd \texttt{fan})$
$\texttt{act3}: \texttt{cdpath} := \{0 \mapsto \texttt{X}\}$
$\texttt{act4}: \texttt{pathl} := 0$
$\texttt{act5}: \texttt{c\_} := \texttt{d\_}$
$\texttt{act6}: \texttt{stage} := 1$

end

Event *extendfan* $\widehat{=}$
Fan is extended first, as long as d_ is not c_ and it can be extended
extends *extendfan*

any

z

where

$\texttt{grd4}: \texttt{l} \in \mathbb{N}_1$
$\texttt{grd1}: \texttt{z} \notin \texttt{ran}(\texttt{fan})$
$\texttt{grd2}: \texttt{X} \mapsto \texttt{z} \in \texttt{dom}(\texttt{coloring2})$
$\texttt{grd3}: \forall \texttt{w} \cdot \forall \texttt{d} \cdot \texttt{fan}(\texttt{l}) \mapsto \texttt{w} \mapsto \texttt{d} \in \texttt{coloring2} \Rightarrow \texttt{d} \neq \texttt{coloring2}(\texttt{X} \mapsto \texttt{z})$
$\texttt{grd5}: \texttt{pathl} = 0$
$\texttt{grd7}: \texttt{stage} = 3$

then

$\texttt{act1}: \texttt{l} := \texttt{l} + 1$
$\texttt{act2}: \texttt{fan} := \texttt{fan} \cup \{\texttt{l} + 1 \mapsto \texttt{z}\}$
$\texttt{act3}: \texttt{d\_} : |\forall \texttt{w} \cdot \texttt{z} \mapsto \texttt{w} \mapsto \texttt{d\_}' \notin \texttt{coloring2}$
We need to update d, the free color of the last node on the fan.

end

Event *extendcdpath* $\widehat{=}$
cd-path is calculated if fan is maximal and c_ is not d_
Status convergent
extends *extendcdpath*

any

z

where

$\texttt{grd1}: \texttt{z} \in \texttt{V}$
$\texttt{grd2}: \texttt{z} \notin \texttt{ran}(\texttt{cdpath})$
$\texttt{grd3}: \texttt{cdpath}(\texttt{pathl}) \mapsto \texttt{z} \mapsto \texttt{c\_} \in \texttt{coloring2} \lor \texttt{cdpath}(\texttt{pathl}) \mapsto \texttt{z} \mapsto \texttt{d\_} \in \texttt{coloring2}$
$\texttt{grd4}: \forall \texttt{z} \cdot \texttt{X} \mapsto \texttt{z} \mapsto \texttt{c\_} \notin \texttt{coloring2}$

grd7 : $\mathtt{stage} = 2$
grd5 : $\mathtt{c\_} \neq \mathtt{d\_}$

then

act1 : $\mathtt{cdpath} := \mathtt{cdpath} \cup \{\mathtt{pathl} + 1 \mapsto \mathtt{z}\}$
act2 : $\mathtt{pathl} := \mathtt{pathl} + 1$

end

Event *fan_done* $\widehat{=}$
Status anticipated
extends *fan_done*

when

grd1 : $\mathtt{l} \in \mathbb{N}_1$
grd2 : $\neg(\exists \mathtt{z}\cdot(\mathtt{z} \notin \mathtt{ran}(\mathtt{fan}) \land \mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom}(\mathtt{coloring2}) \land \forall \mathtt{w}\cdot\forall \mathtt{d}\cdot\mathtt{fan}(\mathtt{l}) \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{z})))$
grd3 : $\mathtt{pathl} = 0$
grd4 : $\mathtt{stage} = 3$

then

act1 : $\mathtt{c\_} :| \forall \mathtt{z}\cdot\mathtt{X} \mapsto \mathtt{z} \mapsto \mathtt{c\_}' \notin \mathtt{coloring2}$
act2 : $\mathtt{stage} := 2$

end

Event *noinvertpath* $\widehat{=}$
Status anticipated
extends *noinvertpath*

when

grd2 : $\mathtt{l} \in \mathbb{N}_1$
grd4 : $\forall \mathtt{z}\cdot\mathtt{X} \mapsto \mathtt{z} \mapsto \mathtt{d\_} \notin \mathtt{coloring2}$
d is free on X, so no path to build.
grd5 : $\neg(\exists \mathtt{z}\cdot(\mathtt{z} \notin \mathtt{ran}(\mathtt{fan}) \land \mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom}(\mathtt{coloring2}) \land (\forall \mathtt{w}\cdot\forall \mathtt{d}\cdot\mathtt{fan}(\mathtt{l}) \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{z}))))$
This ensures that the fan is maximal.
grd1 : $\mathtt{stage} = 2$

then

act4 : $\mathtt{c\_} := \mathtt{d\_}$
If the fan is maximal and d free on X, then the cd-path is empty and we conclude that d is free on X.
act1 : $\mathtt{stage} := 1$

end

VARIANT

$\mathtt{min}(\{\mathtt{max}(\{\mathtt{stage} - 1, 0\}), 1\}) * (1 + \mathtt{card}(\mathtt{V} \setminus \mathtt{ran}(\mathtt{cdpath})))$
The variant is zero in stage 3 and otherwise decreases while the size of the cd-path increases.

END

A.12. m10 Conv Fan Color

MACHINE m10_Conv_Fan_Color
This refinement shows the convergence of fan destruction.

REFINES m09_Conv_CDpath
SEES Input
VARIABLES

coloring

coloring2

X

Y

fan

l   length of fan

cdpath   cd-path (or prefix while building) (I am running ot of names)

pathl   length of cd path

d_

c_

stage   stage 3: buiding fan, stage 2: building cd-path and inverting it, stage 1: moving color along path

EVENTS
Initialisation

extended

begin

act1 : $\mathtt{coloring} := \varnothing$
act2 : $\mathtt{coloring2} := \varnothing$
act3 : $\mathtt{X, Y, fan, cdpath} : |\mathtt{X}' \mapsto \mathtt{Y}' \in \mathtt{graph} \wedge \mathtt{fan}' = \{1 \mapsto \mathtt{Y}'\} \wedge \mathtt{cdpath}' = \{0 \mapsto \mathtt{X}'\}$
act4 : $\mathtt{l} := 1$
act5 : $\mathtt{pathl} := 0$
act6 : $\mathtt{c\_} :\in \mathtt{C}$
act7 : $\mathtt{d\_} :\in \mathtt{C}$
act8 : $\mathtt{stage} := 3$

end

Event *finish* $\widehat{=}$
extends *finish*

when

grd1 : $\mathtt{dom(coloring2)} = \mathtt{graph}$

then

act1 : $\mathtt{coloring} := \mathtt{coloring2}$

end

Event *color1a* $\widehat{=}$
extends *color1a*

any

d

where

grd1 : $\mathtt{X} \mapsto \mathtt{Y} \in \mathtt{graph}$
grd2 : $\mathtt{X} \mapsto \mathtt{Y} \notin \mathtt{dom(coloring2)}$
grd3 : $\forall \mathtt{z} \cdot (\mathtt{X} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$
grd4 : $\forall \mathtt{z} \cdot (\mathtt{Y} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$

grd5 : $\mathtt{dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}) = graph}$
This is the last uncolored edge.
grd6 : $\mathtt{pathl = 0}$
grd7 : $\mathtt{stage = 1}$

then

act1 : $\mathtt{coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}}$
act4 : $\mathtt{fan := \varnothing}$
We remove the fan.
act3 : $\mathtt{l := 0}$
act5 : $\mathtt{stage := 3}$

end

Event *color1b* $\widehat{=}$
extends *color1b*

any

d
X2 So here is our next edge to be colored.
Y2

where

grd1 : $\mathtt{X \mapsto Y \in graph}$
grd2 : $\mathtt{X \mapsto Y \notin dom(coloring2)}$
grd3 : $\mathtt{\forall z \cdot (X \mapsto z) \mapsto d \notin coloring2}$
grd4 : $\mathtt{\forall z \cdot (Y \mapsto z) \mapsto d \notin coloring2}$
grd5 : $\mathtt{X2 \mapsto Y2 \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})}$
grd6 : $\mathtt{stage = 1}$

then

act1 : $\mathtt{coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}}$
act2 : $\mathtt{X := X2}$
act3 : $\mathtt{Y := Y2}$
act4 : $\mathtt{fan := \{1 \mapsto Y2\}}$
The fan is initialized with the new value of Y.
act5 : $\mathtt{l := 1}$
act6 : $\mathtt{cdpath := \{0 \mapsto X2\}}$
act7 : $\mathtt{pathl := 0}$
act8 : $\mathtt{d\_ : | \forall z \cdot (Y2 \mapsto z \mapsto d\_') \notin coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}}$
We need to find a new d.
act9 : $\mathtt{stage := 3}$

end

Event *colormove* $\widehat{=}$
Status convergent
extends *colormove*

when

grd1 : $\mathtt{X \mapsto Y \in graph}$
grd3 : $\mathtt{X \mapsto Y \notin dom(coloring2)}$
grd4 : $\mathtt{l \geq 2}$
grd5 : $\mathtt{pathl = 0}$
grd6 : $\mathtt{stage = 1}$

then

act1 : $\texttt{coloring2} := (\{\texttt{X} \mapsto \texttt{fan}(2), \texttt{fan}(2) \mapsto \texttt{X}\} \lhd \texttt{coloring2}) \cup \{\texttt{X} \mapsto \texttt{Y} \mapsto \texttt{coloring2}(\texttt{X} \mapsto \texttt{fan}(2)), \texttt{Y} \mapsto \texttt{X} \mapsto \texttt{coloring2}(\texttt{X} \mapsto \texttt{fan}(2))\}$
act2 : $\texttt{Y} := \texttt{fan}(2)$
act3 : $\texttt{fan} := (\lambda \texttt{n} \cdot \texttt{n} \in 1 \,..\, \texttt{l} - 1 | \texttt{fan}(\texttt{n} + 1))$
The fan is updated afterwards.
act4 : $\texttt{l} := \texttt{l} - 1$
end

Event *invertpath_k* $\widehat{=}$
Status anticipated
extends *invertpath_k*
any
k Fan edge with color d
where
grd16 : $\texttt{pathl} > 0$
grd9 : $\texttt{l} \in \mathbb{N}_1$
grd11 : $\texttt{k} \in 1 \,..\, \texttt{l} - 1$
grd10 : $(\texttt{X} \mapsto \texttt{fan}(\texttt{k} + 1) \mapsto \texttt{d\_}) \in \texttt{coloring2}$
grd15 : $\forall \texttt{z} \cdot (\texttt{cdpath}(\texttt{pathl}) \mapsto \texttt{z} \mapsto \texttt{c\_} \in \texttt{coloring2} \lor \texttt{cdpath}(\texttt{pathl}) \mapsto \texttt{z} \mapsto \texttt{d\_} \in \texttt{coloring2}) \Rightarrow (\texttt{z} = \texttt{cdpath}(\texttt{pathl} - 1))$
This ensures that the cd-path is already fully calculated.
grd17 : $\texttt{stage} = 2$
then
act1 : $\texttt{coloring2} := \{(\texttt{y} \mapsto \texttt{z} \mapsto \texttt{e}') | \exists \texttt{e} \cdot (\texttt{y} \mapsto \texttt{z} \mapsto \texttt{e}) \in \texttt{coloring2} \land (((\texttt{y} \in \texttt{ran}(\texttt{cdpath}) \lor \texttt{z} \in \texttt{ran}(\texttt{cdpath})) \land ((\texttt{e} = \texttt{d\_} \land \texttt{e}' = \texttt{c\_}) \lor (\texttt{e} = \texttt{c\_} \land \texttt{e}' = \texttt{d\_}) \lor (\texttt{e} \neq \texttt{d\_} \land \texttt{e} \neq \texttt{c\_} \land \texttt{e} = \texttt{e}'))) \lor (\texttt{y} \notin \texttt{ran}(\texttt{cdpath}) \land \texttt{z} \notin \texttt{ran}(\texttt{cdpath}) \land \texttt{e}' = \texttt{e}))\}$
act2 : $\texttt{l}, \texttt{fan} : |(\texttt{fan}(\texttt{k}) \in \texttt{ran}(\texttt{cdpath}) \Rightarrow \texttt{l}' = \texttt{l} \land \texttt{fan}' = \texttt{fan}) \land (\texttt{fan}(\texttt{k}) \notin \texttt{ran}(\texttt{cdpath}) \Rightarrow \texttt{l}' = \texttt{k} \land \texttt{fan}' = 1 \,..\, \texttt{k} \lhd \texttt{fan})$
act3 : $\texttt{cdpath} := \{0 \mapsto \texttt{X}\}$
act4 : $\texttt{pathl} := 0$
act5 : $\texttt{c\_} := \texttt{d\_}$
act6 : $\texttt{stage} := 1$
end

Event *extendfan* $\widehat{=}$
Fan is extended first, as long as d_ is not c_ and it can be extended
extends *extendfan*
any
z
where
grd4 : $\texttt{l} \in \mathbb{N}_1$
grd1 : $\texttt{z} \notin \texttt{ran}(\texttt{fan})$
grd2 : $\texttt{X} \mapsto \texttt{z} \in \texttt{dom}(\texttt{coloring2})$
grd3 : $\forall \texttt{w} \cdot \forall \texttt{d} \cdot \texttt{fan}(\texttt{l}) \mapsto \texttt{w} \mapsto \texttt{d} \in \texttt{coloring2} \Rightarrow \texttt{d} \neq \texttt{coloring2}(\texttt{X} \mapsto \texttt{z})$
grd5 : $\texttt{pathl} = 0$
grd7 : $\texttt{stage} = 3$
then
act1 : $\texttt{l} := \texttt{l} + 1$
act2 : $\texttt{fan} := \texttt{fan} \cup \{\texttt{l} + 1 \mapsto \texttt{z}\}$

act3 : $d\_ :| \forall w \cdot z \mapsto w \mapsto d\_' \notin coloring2$
We need to update d, the free color of the last node on the fan.

end

Event *extendcdpath* $\widehat{=}$
cd-path is calculated if fan is maximal and c_ is not d_
extends *extendcdpath*

any
z
where
grd1 : $z \in V$
grd2 : $z \notin ran(cdpath)$
grd3 : $cdpath(pathl) \mapsto z \mapsto c\_ \in coloring2 \lor cdpath(pathl) \mapsto z \mapsto d\_ \in coloring2$
grd4 : $\forall z \cdot X \mapsto z \mapsto c\_ \notin coloring2$
grd7 : $stage = 2$
grd5 : $c\_ \neq d\_$
then
act1 : $cdpath := cdpath \cup \{pathl + 1 \mapsto z\}$
act2 : $pathl := pathl + 1$
end

Event *fan_done* $\widehat{=}$
Status anticipated
extends *fan_done*

when
grd1 : $l \in \mathbb{N}_1$
grd2 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land \forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z)))$
grd3 : $pathl = 0$
grd4 : $stage = 3$
then
act1 : $c\_ :| \forall z \cdot X \mapsto z \mapsto c\_' \notin coloring2$
act2 : $stage := 2$
end

Event *noinvertpath* $\widehat{=}$
Status anticipated
extends *noinvertpath*

when
grd2 : $l \in \mathbb{N}_1$
grd4 : $\forall z \cdot X \mapsto z \mapsto d\_ \notin coloring2$
d is free on X, so no path to build.
grd5 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land (\forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z))))$
This ensures that the fan is maximal.
grd1 : $stage = 2$
then

act4 : $c\_ := d\_$
If the fan is maximal and d free on X, then the cd-path is empty and we conclude that d is free on X.
act1 : $stage := 1$
end

VARIANT

$(min(\{stage, 2\}) - 1) * (card(V) + 1) + max(\{2 - stage, 0\}) * (1 + card(ran(fan)))$
In stage 3 and 2, this is set to a large constant. In stage 1 it decreases with the size of the fan.

END

A.13. m11 Conv Stages

MACHINE m11_Conv_Stages
To combine the three previous convergence proofs, we show that the stage variable decreases.
REFINES m10_Conv_Fan_Color
SEES Input
VARIABLES

coloring
coloring2
X
Y
fan
l length of fan
cdpath cd-path (or prefix while building) (I am running ot of names)
pathl length of cd path
d_
c_
stage stage 3: buiding fan, stage 2: building cd-path and inverting it, stage 1: moving color along path

EVENTS
Initialisation
extended
begin
act1 : $coloring := \varnothing$
act2 : $coloring2 := \varnothing$
act3 : $X, Y, fan, cdpath : |X' \mapsto Y' \in graph \land fan' = \{1 \mapsto Y'\} \land cdpath' = \{0 \mapsto X'\}$
act4 : $l := 1$
act5 : $pathl := 0$
act6 : $c\_ :\in C$
act7 : $d\_ :\in C$
act8 : $stage := 3$
end

Event *finish* $\widehat{=}$
extends *finish*

when

grd1 : dom(coloring2) = graph

then

act1 : coloring := coloring2

end

Event *color1a* $\widehat{=}$
extends *color1a*

any

d

where

grd1 : X ↦ Y ∈ graph

grd2 : X ↦ Y ∉ dom(coloring2)

grd3 : ∀z·(X ↦ z) ↦ d ∉ coloring2

grd4 : ∀z·(Y ↦ z) ↦ d ∉ coloring2

grd5 : dom(coloring2 ∪ {X ↦ Y ↦ d, Y ↦ X ↦ d}) = graph

This is the last uncolored edge.

grd6 : pathl = 0

grd7 : stage = 1

then

act1 : coloring2 := coloring2 ∪ {X ↦ Y ↦ d, Y ↦ X ↦ d}

act4 : fan := ∅

We remove the fan.

act3 : l := 0

act5 : stage := 3

end

Event *color1b* $\widehat{=}$
extends *color1b*

any

d

X2 So here is our next edge to be colored.

Y2

where

grd1 : X ↦ Y ∈ graph

grd2 : X ↦ Y ∉ dom(coloring2)

grd3 : ∀z·(X ↦ z) ↦ d ∉ coloring2

grd4 : ∀z·(Y ↦ z) ↦ d ∉ coloring2

grd5 : X2 ↦ Y2 ∈ graph \ dom(coloring2 ∪ {X ↦ Y ↦ d, Y ↦ X ↦ d})

grd6 : stage = 1

then

act1 : coloring2 := coloring2 ∪ {X ↦ Y ↦ d, Y ↦ X ↦ d}

act2 : X := X2

act3 : Y := Y2

act4 : fan := {1 ↦ Y2}

The fan is initialized with the new value of Y.

act5 : $\mathtt{l} := 1$
act6 : $\mathtt{cdpath} := \{0 \mapsto \mathtt{X2}\}$
act7 : $\mathtt{pathl} := 0$
act8 : $\mathtt{d\_} : |\forall \mathtt{z}\cdot(\mathtt{Y2} \mapsto \mathtt{z} \mapsto \mathtt{d\_}') \notin \mathtt{coloring2} \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{d}, \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{d}\}$
We need to find a new d.
act9 : $\mathtt{stage} := 3$

end

Event *colormove* $\widehat{=}$
extends *colormove*

when
grd1 : $\mathtt{X} \mapsto \mathtt{Y} \in \mathtt{graph}$
grd3 : $\mathtt{X} \mapsto \mathtt{Y} \notin \mathrm{dom}(\mathtt{coloring2})$
grd4 : $\mathtt{l} \geq 2$
grd5 : $\mathtt{pathl} = 0$
grd6 : $\mathtt{stage} = 1$

then
act1 : $\mathtt{coloring2} := (\{\mathtt{X} \mapsto \mathtt{fan}(2), \mathtt{fan}(2) \mapsto \mathtt{X}\} \lhd\!\!\!- \mathtt{coloring2}) \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{fan}(2)), \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{fan}(2))\}$
act2 : $\mathtt{Y} := \mathtt{fan}(2)$
act3 : $\mathtt{fan} := (\lambda \mathtt{n}\cdot\mathtt{n} \in 1 \,..\, \mathtt{l} - 1 | \mathtt{fan}(\mathtt{n} + 1))$
The fan is updated afterwards.
act4 : $\mathtt{l} := \mathtt{l} - 1$

end

Event *invertpath_k* $\widehat{=}$
Status convergent
extends *invertpath_k*

any
k   Fan edge with color d

where
grd16 : $\mathtt{pathl} > 0$
grd9 : $\mathtt{l} \in \mathbb{N}_1$
grd11 : $\mathtt{k} \in 1 \,..\, \mathtt{l} - 1$
grd10 : $(\mathtt{X} \mapsto \mathtt{fan}(\mathtt{k} + 1) \mapsto \mathtt{d\_}) \in \mathtt{coloring2}$
grd15 : $\forall \mathtt{z}\cdot(\mathtt{cdpath}(\mathtt{pathl}) \mapsto \mathtt{z} \mapsto \mathtt{c\_} \in \mathtt{coloring2} \vee \mathtt{cdpath}(\mathtt{pathl}) \mapsto \mathtt{z} \mapsto \mathtt{d\_} \in \mathtt{coloring2}) \Rightarrow (\mathtt{z} = \mathtt{cdpath}(\mathtt{pathl} - 1))$
This ensures that the cd-path is already fully calculated.
grd17 : $\mathtt{stage} = 2$

then
act1 : $\mathtt{coloring2} := \{(\mathtt{y} \mapsto \mathtt{z} \mapsto \mathtt{e}')|\exists \mathtt{e}\cdot(\mathtt{y} \mapsto \mathtt{z} \mapsto \mathtt{e}) \in \mathtt{coloring2} \wedge (((\mathtt{y} \in \mathrm{ran}(\mathtt{cdpath}) \vee \mathtt{z} \in \mathrm{ran}(\mathtt{cdpath})) \wedge ((\mathtt{e} = \mathtt{d\_} \wedge \mathtt{e}' = \mathtt{c\_}) \vee (\mathtt{e} = \mathtt{c\_} \wedge \mathtt{e}' = \mathtt{d\_}) \vee (\mathtt{e} \neq \mathtt{d\_} \wedge \mathtt{e} \neq \mathtt{c\_} \wedge \mathtt{e} = \mathtt{e}'))) \vee (\mathtt{y} \notin \mathrm{ran}(\mathtt{cdpath}) \wedge \mathtt{z} \notin \mathrm{ran}(\mathtt{cdpath}) \wedge \mathtt{e}' = \mathtt{e}))\}$
act2 : $\mathtt{l}, \mathtt{fan} : |(\mathtt{fan}(\mathtt{k}) \in \mathrm{ran}(\mathtt{cdpath}) \Rightarrow \mathtt{l}' = \mathtt{l} \wedge \mathtt{fan}' = \mathtt{fan}) \wedge (\mathtt{fan}(\mathtt{k}) \notin \mathrm{ran}(\mathtt{cdpath}) \Rightarrow \mathtt{l}' = \mathtt{k} \wedge \mathtt{fan}' = 1 \,..\, \mathtt{k} \lhd \mathtt{fan})$
act3 : $\mathtt{cdpath} := \{0 \mapsto \mathtt{X}\}$
act4 : $\mathtt{pathl} := 0$
act5 : $\mathtt{c\_} := \mathtt{d\_}$
act6 : $\mathtt{stage} := 1$

end

Event *extendfan* $\widehat{=}$
Fan is extended first, as long as d_ is not c_ and it can be extended
extends *extendfan*

any
z
where
grd4 : $\mathtt{l} \in \mathbb{N}_1$
grd1 : $\mathtt{z} \notin \mathtt{ran(fan)}$
grd2 : $\mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom(coloring2)}$
grd3 : $\forall \mathtt{w} \cdot \forall \mathtt{d} \cdot \mathtt{fan(l)} \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2(X} \mapsto \mathtt{z)}$
grd5 : $\mathtt{pathl} = 0$
grd7 : $\mathtt{stage} = 3$
then
act1 : $\mathtt{l} := \mathtt{l} + 1$
act2 : $\mathtt{fan} := \mathtt{fan} \cup \{\mathtt{l} + 1 \mapsto \mathtt{z}\}$
act3 : $\mathtt{d\_} :| \forall \mathtt{w} \cdot \mathtt{z} \mapsto \mathtt{w} \mapsto \mathtt{d\_}' \notin \mathtt{coloring2}$
We need to update d, the free color of the last node on the fan.
end

Event *extendcdpath* $\widehat{=}$
cd-path is calculated if fan is maximal and c_ is not d_
extends *extendcdpath*

any
z
where
grd1 : $\mathtt{z} \in \mathtt{V}$
grd2 : $\mathtt{z} \notin \mathtt{ran(cdpath)}$
grd3 : $\mathtt{cdpath(pathl)} \mapsto \mathtt{z} \mapsto \mathtt{c\_} \in \mathtt{coloring2} \lor \mathtt{cdpath(pathl)} \mapsto \mathtt{z} \mapsto \mathtt{d\_} \in \mathtt{coloring2}$
grd4 : $\forall \mathtt{z} \cdot \mathtt{X} \mapsto \mathtt{z} \mapsto \mathtt{c\_} \notin \mathtt{coloring2}$
grd7 : $\mathtt{stage} = 2$
grd5 : $\mathtt{c\_} \neq \mathtt{d\_}$
then
act1 : $\mathtt{cdpath} := \mathtt{cdpath} \cup \{\mathtt{pathl} + 1 \mapsto \mathtt{z}\}$
act2 : $\mathtt{pathl} := \mathtt{pathl} + 1$
end

Event *fan_done* $\widehat{=}$
Status convergent
extends *fan_done*

when
grd1 : $\mathtt{l} \in \mathbb{N}_1$
grd2 : $\neg(\exists \mathtt{z} \cdot (\mathtt{z} \notin \mathtt{ran(fan)} \land \mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom(coloring2)} \land \forall \mathtt{w} \cdot \forall \mathtt{d} \cdot \mathtt{fan(l)} \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2(X} \mapsto \mathtt{z)}))$
grd3 : $\mathtt{pathl} = 0$
grd4 : $\mathtt{stage} = 3$
then

act1 : $c\_ :| \forall z \cdot X \mapsto z \mapsto c\_' \notin coloring2$
act2 : $stage := 2$

end

Event *noinvertpath* $\widehat{=}$
Status convergent
extends *noinvertpath*

when

grd2 : $l \in \mathbb{N}_1$
grd4 : $\forall z \cdot X \mapsto z \mapsto d\_ \notin coloring2$
d is free on X, so no path to build.
grd5 : $\neg(\exists z \cdot (z \notin ran(fan) \land X \mapsto z \in dom(coloring2) \land (\forall w \cdot \forall d \cdot fan(l) \mapsto w \mapsto d \in coloring2 \Rightarrow d \neq coloring2(X \mapsto z))))$
This ensures that the fan is maximal.
grd1 : $stage = 2$

then

act4 : $c\_ := d\_$
If the fan is maximal and d free on X, then the cd-path is empty and we conclude that d is free on X.
act1 : $stage := 1$

end

VARIANT

stage

END

A.14. m12 Deadlock Freedom

MACHINE m12_Deadlock_Freedom

The last refinement has no visible changes in this report. Nevertheless, it has one proof obligation: The deadlock freedom, which is the disjunction of all event guards. By discharging this, we verify that our program does eventually reach the desired final state.

REFINES m11_Conv_Stages
SEES Input
VARIABLES

coloring

coloring2

X

Y

fan

l length of fan

cdpath cd-path (or prefix while building) (I am running ot of names)

pathl length of cd path

d_

c_

`stage` stage 3: buiding fan, stage 2: building cd-path and inverting it, stage 1: moving color along path

EVENTS
Initialisation
extended
begin
act1 : $\mathtt{coloring} := \varnothing$
act2 : $\mathtt{coloring2} := \varnothing$
act3 : $\mathtt{X, Y, fan, cdpath} : |\mathtt{X}' \mapsto \mathtt{Y}' \in \mathtt{graph} \land \mathtt{fan}' = \{1 \mapsto \mathtt{Y}'\} \land \mathtt{cdpath}' = \{0 \mapsto \mathtt{X}'\}$
act4 : $\mathtt{l} := 1$
act5 : $\mathtt{pathl} := 0$
act6 : $\mathtt{c\_} :\in \mathtt{C}$
act7 : $\mathtt{d\_} :\in \mathtt{C}$
act8 : $\mathtt{stage} := 3$
end

Event *finish* $\widehat{=}$
extends *finish*
when
grd1 : $\mathtt{dom(coloring2) = graph}$
then
act1 : $\mathtt{coloring} := \mathtt{coloring2}$
end

Event *color1a* $\widehat{=}$
extends *color1a*
any
d
where
grd1 : $\mathtt{X} \mapsto \mathtt{Y} \in \mathtt{graph}$
grd2 : $\mathtt{X} \mapsto \mathtt{Y} \notin \mathtt{dom(coloring2)}$
grd3 : $\forall \mathtt{z} \cdot (\mathtt{X} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$
grd4 : $\forall \mathtt{z} \cdot (\mathtt{Y} \mapsto \mathtt{z}) \mapsto \mathtt{d} \notin \mathtt{coloring2}$
grd5 : $\mathtt{dom}(\mathtt{coloring2} \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{d}, \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{d}\}) = \mathtt{graph}$
This is the last uncolored edge.
grd6 : $\mathtt{pathl} = 0$
grd7 : $\mathtt{stage} = 1$
then
act1 : $\mathtt{coloring2} := \mathtt{coloring2} \cup \{\mathtt{X} \mapsto \mathtt{Y} \mapsto \mathtt{d}, \mathtt{Y} \mapsto \mathtt{X} \mapsto \mathtt{d}\}$
act4 : $\mathtt{fan} := \varnothing$
We remove the fan.
act3 : $\mathtt{l} := 0$
act5 : $\mathtt{stage} := 3$
end

Event *color1b* $\widehat{=}$
extends *color1b*
any
d

X2 So here is our next edge to be colored.
Y2

where

grd1 : $X \mapsto Y \in graph$
grd2 : $X \mapsto Y \notin dom(coloring2)$
grd3 : $\forall z\cdot(X \mapsto z) \mapsto d \notin coloring2$
grd4 : $\forall z\cdot(Y \mapsto z) \mapsto d \notin coloring2$
grd5 : $X2 \mapsto Y2 \in graph \setminus dom(coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\})$
grd6 : $stage = 1$

then

act1 : $coloring2 := coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
act2 : $X := X2$
act3 : $Y := Y2$
act4 : $fan := \{1 \mapsto Y2\}$
The fan is initialized with the new value of Y.
act5 : $l := 1$
act6 : $cdpath := \{0 \mapsto X2\}$
act7 : $pathl := 0$
act8 : $d\_ : |\forall z\cdot(Y2 \mapsto z \mapsto d\_') \notin coloring2 \cup \{X \mapsto Y \mapsto d, Y \mapsto X \mapsto d\}$
We need to find a new d.
act9 : $stage := 3$

end

Event *colormove* $\widehat{=}$
extends *colormove*

when

grd1 : $X \mapsto Y \in graph$
grd3 : $X \mapsto Y \notin dom(coloring2)$
grd4 : $l \geq 2$
grd5 : $pathl = 0$
grd6 : $stage = 1$

then

act1 : $coloring2 := (\{X \mapsto fan(2), fan(2) \mapsto X\} \lhd coloring2) \cup \{X \mapsto Y \mapsto coloring2(X \mapsto fan(2)), Y \mapsto X \mapsto coloring2(X \mapsto fan(2))\}$
act2 : $Y := fan(2)$
act3 : $fan := (\lambda n\cdot n \in 1\,..\,l-1|fan(n+1))$
The fan is updated afterwards.
act4 : $l := l - 1$

end

Event *invertpath_k* $\widehat{=}$
extends *invertpath_k*

any

k Fan edge with color d

where

grd16 : $pathl > 0$
grd9 : $l \in \mathbb{N}_1$
grd11 : $k \in 1\,..\,l-1$
grd10 : $(X \mapsto fan(k+1) \mapsto d\_) \in coloring2$

grd15 : $\forall z\cdot(\mathtt{cdpath}(\mathtt{pathl}) \mapsto z \mapsto c\_ \in \mathtt{coloring2} \lor \mathtt{cdpath}(\mathtt{pathl}) \mapsto z \mapsto d\_ \in \mathtt{coloring2}) \Rightarrow (z = \mathtt{cdpath}(\mathtt{pathl} - 1))$
This ensures that the cd-path is already fully calculated.
grd17 : $\mathtt{stage} = 2$

then

act1 : $\mathtt{coloring2} := \{(y \mapsto z \mapsto e')|\exists e\cdot(y \mapsto z \mapsto e) \in \mathtt{coloring2} \land (((y \in \mathtt{ran}(\mathtt{cdpath}) \lor z \in \mathtt{ran}(\mathtt{cdpath})) \land ((e = d\_ \land e' = c\_) \lor (e = c\_ \land e' = d\_) \lor (e \neq d\_ \land e \neq c\_ \land e = e'))) \lor (y \notin \mathtt{ran}(\mathtt{cdpath}) \land z \notin \mathtt{ran}(\mathtt{cdpath}) \land e' = e))\}$
act2 : $\mathtt{l}, \mathtt{fan} :| (\mathtt{fan}(k) \in \mathtt{ran}(\mathtt{cdpath}) \Rightarrow \mathtt{l}' = \mathtt{l} \land \mathtt{fan}' = \mathtt{fan}) \land (\mathtt{fan}(k) \notin \mathtt{ran}(\mathtt{cdpath}) \Rightarrow \mathtt{l}' = k \land \mathtt{fan}' = 1\,..\,k \lhd \mathtt{fan})$
act3 : $\mathtt{cdpath} := \{0 \mapsto X\}$
act4 : $\mathtt{pathl} := 0$
act5 : $c\_ := d\_$
act6 : $\mathtt{stage} := 1$

end

Event *extendfan* $\widehat{=}$
Fan is extended first, as long as d_ is not c_ and it can be extended
extends *extendfan*

any
z
where

grd4 : $\mathtt{l} \in \mathbb{N}_1$
grd1 : $z \notin \mathtt{ran}(\mathtt{fan})$
grd2 : $X \mapsto z \in \mathtt{dom}(\mathtt{coloring2})$
grd3 : $\forall w\cdot\forall d\cdot \mathtt{fan}(\mathtt{l}) \mapsto w \mapsto d \in \mathtt{coloring2} \Rightarrow d \neq \mathtt{coloring2}(X \mapsto z)$
grd5 : $\mathtt{pathl} = 0$
grd7 : $\mathtt{stage} = 3$

then

act1 : $\mathtt{l} := \mathtt{l} + 1$
act2 : $\mathtt{fan} := \mathtt{fan} \cup \{\mathtt{l} + 1 \mapsto z\}$
act3 : $d\_ :| \forall w\cdot z \mapsto w \mapsto d\_' \notin \mathtt{coloring2}$
We need to update d, the free color of the last node on the fan.

end

Event *extendcdpath* $\widehat{=}$
cd-path is calculated if fan is maximal and c_ is not d_
extends *extendcdpath*

any
z
where

grd1 : $z \in V$
grd2 : $z \notin \mathtt{ran}(\mathtt{cdpath})$
grd3 : $\mathtt{cdpath}(\mathtt{pathl}) \mapsto z \mapsto c\_ \in \mathtt{coloring2} \lor \mathtt{cdpath}(\mathtt{pathl}) \mapsto z \mapsto d\_ \in \mathtt{coloring2}$
grd4 : $\forall z\cdot X \mapsto z \mapsto c\_ \notin \mathtt{coloring2}$
grd7 : $\mathtt{stage} = 2$
grd5 : $c\_ \neq d\_$

then

act1 : $\mathtt{cdpath} := \mathtt{cdpath} \cup \{\mathtt{pathl} + 1 \mapsto \mathtt{z}\}$
act2 : $\mathtt{pathl} := \mathtt{pathl} + 1$

end

Event *fan_done* $\widehat{=}$
extends *fan_done*

when

grd1 : $\mathtt{l} \in \mathbb{N}_1$
grd2 : $\neg(\exists \mathtt{z} \cdot (\mathtt{z} \notin \mathtt{ran}(\mathtt{fan}) \land \mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom}(\mathtt{coloring2}) \land \forall \mathtt{w} \cdot \forall \mathtt{d} \cdot \mathtt{fan}(\mathtt{l}) \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{z})))$
grd3 : $\mathtt{pathl} = 0$
grd4 : $\mathtt{stage} = 3$

then

act1 : $\mathtt{c\_} :| \forall \mathtt{z} \cdot \mathtt{X} \mapsto \mathtt{z} \mapsto \mathtt{c\_}' \notin \mathtt{coloring2}$
act2 : $\mathtt{stage} := 2$

end

Event *noinvertpath* $\widehat{=}$
extends *noinvertpath*

when

grd2 : $\mathtt{l} \in \mathbb{N}_1$
grd4 : $\forall \mathtt{z} \cdot \mathtt{X} \mapsto \mathtt{z} \mapsto \mathtt{d\_} \notin \mathtt{coloring2}$
d is free on X, so no path to build.
grd5 : $\neg(\exists \mathtt{z} \cdot (\mathtt{z} \notin \mathtt{ran}(\mathtt{fan}) \land \mathtt{X} \mapsto \mathtt{z} \in \mathtt{dom}(\mathtt{coloring2}) \land (\forall \mathtt{w} \cdot \forall \mathtt{d} \cdot \mathtt{fan}(\mathtt{l}) \mapsto \mathtt{w} \mapsto \mathtt{d} \in \mathtt{coloring2} \Rightarrow \mathtt{d} \neq \mathtt{coloring2}(\mathtt{X} \mapsto \mathtt{z}))))$
This ensures that the fan is maximal.
grd1 : $\mathtt{stage} = 2$

then

act4 : $\mathtt{c\_} := \mathtt{d\_}$
If the fan is maximal and d free on X, then the cd-path is empty and we conclude that d is free on X.
act1 : $\mathtt{stage} := 1$

end

END