Defining Liveness

by Bowen Alpern and Fred B. Schneider

Presented by Joe Melnyk

Introduction

- view of concurrent program execution
  - a sequence $\sigma = s_0 s_1 s_2 \ldots$ of states
  - each state s_i (for $i > 0$) is the result of a single atomic action from s_{i-1}
  - *property* = set of such sequences
    - a property *P holds* for a program if the set of all sequences defined by the program is contained within the property
- arguments to prove a program satisfies a given property:
  - *safety property* – invariance
  - *liveness property* – well-foundedness

Safety Properties

- informal definition: no “bad things” happen during program execution
- examples and their respective “bad things”
  - mutual exclusion; two processes executing in the critical section at the same time
  - deadlock freedom; deadlock
  - partial correctness; starting state satisfied the precondition, but the termination state does not satisfy the postcondition
  - first-come-first-serve; servicing a request made after one that has not yet been serviced
- formal definition:
  - assumptions
    - let
      - S = set of program states
      - S^ω = set of infinite sequences of program states
      - S^* = set of finite sequences of program states

- execution of a program can be modeled as a member of S^ω
- elements of S^ω = executions
- elements of S^* = partial executions
- $\sigma \models P$ if σ is in property P
- let σ_i = partial execution consisting of the first i states in σ

- in order for P to be a safety property, if P doesn't hold for an execution then a "bad thing" must happen at some point
- the "bad thing" is irremediable since a safety property states that "bad things" never happen during execution
- therefore, P is a *safety property* if and only if
  - $(\forall \sigma: \sigma \in S^\omega: \sigma \not\models P \Rightarrow (\exists i : 0 \leq i: (\forall \beta: \beta \in S^\omega: \sigma_i\beta \not\models P)))$
- by the definition, a safety property unconditionally prohibits a "bad thing" from occurring; if it does occur, there is an identifiable point at which this can be recognized

Liveness Properties

- informal definition: a “good thing” happens during program execution
- examples and their respective “good things”
  - starvation freedom; making progress
  - termination; completion of the final instruction
  - guaranteed service; receiving service
- defining characteristic of liveness
  - no partial execution is irremediable; a “good thing” can always occur in the future
  - note: if a partial execution were irremediable, it would be a “bad thing” and liveness properties cannot reject “bad things”, only ensure “good things”

- formal definition:
  - a partial execution α is *live* for a property *P* if and only if there is a sequence of states β such that $\alpha\beta \models P$
  - in a *liveness property*, every partial execution is live
  - therefore, *P* is a liveness property if and only if

    $(\forall\alpha: \alpha \in S^*: (\exists\beta: \beta \in S^\omega: \alpha\beta \models P)$
  - notice:
    - no restriction on what the "good thing" is nor requirement that it be discrete
      - for example, the "good thing" in starvation freedom (progress) is an infinite collection of discrete events
      - hence, "good things" are fundamentally different from "bad things"
    - a liveness property cannot stipulate that a "good thing" always happens, only that it eventually happens

- the authors believe no liveness definition is more permissive
  - proof (by contradiction):
    - suppose that *P* is a liveness property that doesn't satisfy the definition; then there must be a partial execution α such that $(\forall \beta: \beta \in S^{\omega}: \alpha\beta \not\models P)$
    - since α is a "bad thing" rejected by *P*, *P* is in part a safety property and not a liveness property
    - this contradicts the assumption of *P* being a liveness property
- more restrictive liveness definitions
  - uniform liveness:

    $(\exists \beta: \beta \in S^{\omega}: (\forall \alpha: \alpha \in S^{*}: \alpha\beta \models P))$
    - *P* is a liveness property if and only if there is a single execution (β) that can be appended to every partial execution (α) so that the resulting sequence is in *P*

- absolute liveness

  $(\exists\gamma: \gamma\in S^{\omega}: \gamma|=P)\wedge(\forall\beta: \beta\in S^{\omega}: \beta|=P \Rightarrow (\forall\alpha: \alpha\in S^{*}: \alpha\beta|=P))$

  - *P* is an absolute-liveness property if and only if it is non-empty and any execution (β) in *P* can be appended to any partial execution (α) to obtain a sequence in *P*

- contrast of definitions
  - liveness: if *any* partial execution α can be extended by *some* execution β so that $\alpha\beta$ is in *L* (choice of β may depend on α), then *L* is a liveness property
  - uniform-liveness: if there is a *single* execution β that extends *all* partial execution α such that $\alpha\beta$ is in *U*, then *U* is a uniform-livness property
  - absolute liveness: if *A* is non-empty and *any* execution β in *A* can be used to extend *all* partial executions α, then *A* is an absolute-liveness property
  - any absolute-liveness property is also a uniform-liveness property and any uniform-liveness property is also a liveness property

- absolute-liveness does not include properties that should be considered liveness
  - *leads-to* - any occurrence of an event of type E_1 is eventually followed by an occurrence of an event of type E_2
    - example: guaranteed service
    - such properties are liveness properties when E_2 is satisfiable (E_2 is the “good thing”)
    - *leads-to* properties are not absolute-liveness properties
      - consider execution β where no event of type E_1 or E_2 occurs
      - *leads-to* holds on β, but appending β to a partial execution consisting of a single event of type E_1 yields and execution that does not satisfy the property

- uniform-liveness does not capture the intuition of liveness either
  - examples
    - *predictive* – if *A* initially holds then after some partial execution *B* always holds; otherwise after some partial execution, *B* never holds
    - *predictive* is a liveness property since it requires a “good thing” to happen: either “always *B*” or “always $\neg$*B*”
    - *predictive* is not a uniform-liveness property since there is no *single* sequence that can extend *all* partial executions

- Other Properties (neither safety nor liveness)
  - *until* – eventually an event of type E_2 will happen; all preceding events are of type E_1
    - this is the intersection of a safety and liveness property
      - safety: "`$\neg$ E_1 before E_2' doesn't happen"
      - liveness: "E_2 eventually happens"
    - total correctness is also the intersection of a safety property and a liveness property: partial correctness and termination, respectively
  - topological overview of S^{ω}:
    - safety properties are the closed sets and liveness properties are the dense sets
      - basic open sets: sets of all executions that share a common prefix
      - open set: union of all basic open sets
      - closed set: complement of an open set
      - dense set: intersects every non-empty open set

- Theorem: every property P is the intersection of a safety and a liveness property
  - proof:
    - let $\bar{P}$ be the smallest safety property containing P and let L be $\neg(\bar{P} - P)$
    - then:

$$L \cap \bar{P} = \neg(\bar{P} - P) \cap \bar{P} = (\neg \bar{P} \cup P) \cap \bar{P}$$
$$= (\neg \bar{P} \cap \bar{P}) \cup (P \cap \bar{P}) = P \cap \bar{P}$$
$$= P$$

    - need to show that L is dense and hence a liveness property (using proof by contradiction):
      - assume there is a non-empty open set O contained in $\neg L$ and thus L is not dense
      - then $O \subseteq (\bar{P} - P)$ and hence $P \subseteq (\bar{P} - O)$
      - $\bar{P} - O$ is closed (and is therefore a safety property) since the intersection of two closed sets is closed
      - this contradicts $\bar{P}$ being the smallest safety property containing P

- corollary:

  if a notation Σ for expressing properties is closed under comlement, intersection and topological closure then any Σ-expressible property is the intersection of a Σ-expressible safety property and a Σ-expressible liveness property

  - therefore, to show that
    - every property *P* expressible in a temporal logic is equivalent to the conjunction of a safety and a liveness property expressed in the logic
    - due to the corollary, we just need to show that the smallest safety property containing *P* is also expressible in the logic

- Theorem: If $|S| > 1$ then any property P is the intersection of two liveness properties
  - proof:
    - $\exists$ states a, $b \in S$ by the hypothesis; let L_a (and L_b) be the set of executions with tails that are an infinite sequence of a's (and b's); both L_a and L_b are liveness properties and $L_a \cap L_b = \phi$
    - $(P \cup L_a) \cap (P \cup L_b) = (P \cap P) \cup (P \cap L_a) \cup (P \cap L_b) \cup (L_a \cap L_b) = P$
    - since the union of any set and a dense set is dense, $P \cup L_a$ and $P \cup L_b$ are liveness properties
- corollary:

  if a notation Σ for expressing properties is closed under intersection and there exists Σ-expressible liveness properties with empty intersection than any Σ-expressible property is the intersection of two Σ-expressible liveness properties

- further notes - using the topological definitions given, it can also be shown that:
  - safety and liveness are closed under Boolean operations
  - safety properties are closed under union and intersection
  - liveness properties are closed only under union
  - neither safety nor liveness is closed under complement
  - S^{ω} is the only property which is closed under safety *and* liveness