• Simple **safety** properties:
  
  - LTL: \( G_p \)
  
  - CTL: \( AG_p \)

• Simple **liveness** properties:

  - LTL: \( F_p \)

  - CTL: \( AF_p \)

  - plus fairness constraints (generalized Büchi Automata)

• Full LTL can be translated to **simple liveness + fairness**
• counter examples to a **safety** property are finite traces
  
  – **radius** is the length of shortest initialized path to an arbitrary state
  
  – radius is a **completeness threshold** for (simple) safety properties
  
  – no longer potential counter example traces have to be checked

• every counter example trace to a liveness property is **lasso** shaped:

  – **diameter** is the length of the longest shortest path between two states

  – **wrong**: completeness threshold for liveness properties is **radius + diameter**
• modulo $n$ (here $n = 4$) counter with an explicit set state:

• radius and diameter both constant, but shortest counter example is of length $n$

• **solution:** use $\neg p$ predicated `radius + diameter`:
  
  – restrict Kripke structure to states in which $\neg p$ holds
  
  – calculate radius and diameter in restricted Kripke structure
• liveness is actually bounded liveness: \( Fp \equiv F_{\leq|S|} p \)

\[
F_{\leq|S|} p \equiv p \lor Xp \lor \ldots \lor X^{\lceil |S| \rceil} p
\]

• brute force expansion needs exponential space for symbolic model checking
  (via the standard Büchi-Automata translation)

• counting translation requires twice the number of state bits
\[ s = \text{original state component} \]

\[ \text{counter} = \lceil \log_2 |S| \rceil \text{-bit counter (}|S| = \text{number of original states}) \]

\[ \text{found} = \text{boolean flag: body of liveness property is satisfied} \]

\[ \text{live} = \text{boolean state bit: found is or was true} \]

\[ G (\text{counter} = |S| \rightarrow \text{live}) \]
State recording translation of liveness to safety

- **s** = original state component
- **l2s_s** = copy of original state component to save a state
- **save** = oracle (new primary input) to control when a state is saved
- **saved** = boolean flag set to true when state has been saved
- **found** = boolean flag: body of liveness property is satisfied
- **live** = boolean state bit: **found** is or was true

\[ G (s = l2s_s \rightarrow live) \]
Example: 2-Bit Counter with Self-Loops

\[ \neq F(s = 3) \]
```plaintext
MODULE main
VAR
  s: {0, 1, 2, 3};
ASSIGN
  init(s) := 0;
  next(s) := case
    s = 0: {1, s};
    s = 1: {2, s};
    s = 2: {3, s};
    s = 3: {0, s};
esac;

-- loop detection part
VAR
  counter: 0..4;
ASSIGN
  init(counter) := 0;
  next(counter) := case
    counter < 4: counter + 1;
    1: counter;
esac;

DEFINE
  looped := counter = 4;

-- property observing part
VAR live: boolean;
DEFINE found := s = 3;
ASSIGN
  init(live) := 0;
  next(live) := live | found;
SPEC AG (looped -> live)
```

```plaintext
MODULE main
VAR
  s: {0, 1, 2, 3};
ASSIGN
  init(s) := 0;
  next(s) := case
    s = 0: {1, s};
    s = 1: {2, s};
    s = 2: {3, s};
    s = 3: {0, s};
esac;

-- loop detection part
VAR
  save: boolean;
  saved: boolean;
  l2s_s: {0, 1, 2, 3};
ASSIGN
  init(saved) := 0;
  next(saved) := on_loop;
  init(l2s_s) := s;
  next(l2s_s) := case
    save & !saved: s;
    1: l2s_s;
esac;

DEFINE
  looped := saved & (s = l2s_s);
  on_loop := save | saved;

-- property observing part
VAR live: boolean;
DEFINE found := s = 3;
ASSIGN
  init(live) := 0;
  next(live) := live | found;
SPEC AG (looped -> live)
```
• both translations are complete

• both translations double the number of state bits (in symbolic model checking)

• both translations may double the number of reachable states (really bad for explicit model checking)

• radius in counting translation may increase exponentially: (in symbolic model checking)

\[ r^{\text{counting}} \geq |S| \]

• radius in state recording translation (optimizations possible):

\[ r^{\text{recording}} \leq \max\{r + 2d + 2, r-p + d-p + 1\} = O(\max\{d, d-p\}) \]
• counter examples found are indeed counter examples (correctness)

• conditions for completeness (modulo reachability):
  – if there is a counter example, then there is also a lasso shaped one
  – each trace visits only finite many states

• examples where it works (state variables $\in \mathbb{IN}$):

\[
I(s) \equiv s \in \mathbb{IN} \quad I(s, b) \equiv s = 0 \land b \in \mathbb{IN}
\]

\[
T(s, s') \equiv s \geq s' \land \text{details}(s, s') \quad T((s, b), (s', b')) \equiv I(s', b') \lor \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad s \leq s' \land b = b' \land \text{details}(s, s')
\]
Conclusion

- completeness threshold is different for liveness and safety
  - predicated diameter instead of ordinary diameter as bound

- finite states: efficient translation of liveness to safety
  - through state-recording translation
  - works in practice for symbolic model checking (e.g. with interpolation)

- infinite states: state recording works for some examples
  - combination of state recording with fairness?
  - can we always (efficiently) translate liveness to safety?