**Course: CS 5392**

**Team Members (SM2):**

Tianyi Gao, Kai Qi, Kaiping Xiong, Huan Wu

**Purpose**

This document describes acceptance test cases for the CTL Model Checker project.

**Scope**

This document contains a complete, cumulative set of test cases used in testing version 1.0 of the CTL Model Checker application.

**Result of Test Cases:** PASS 🗹 FAIL □ (Provide comments for failure)

**Comments:**

|  |
| --- |
|  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test #** | |  |  |  | | --- | --- | --- | | **Actions** | **Expected** | **Attachment (Witness)** | | **Results** |
| T1 | |  |  |  | | --- | --- | --- | | **This test demonstrates that the system checks for validness of Kripke Structure input file and StateID/CTL formula to be checked.** | | | | 1. Open application. | -- | -- | | 1. Browse and load “Broken Model 1.txt” | Console should pop up informing that “Input file does not contain appropriate segments to construct kripke structure”. | **T1.2** | | 1. Browse and load “Broken Model 2.txt”. | A message box should pop up informing that “State s1 is defined more than once”. | **T1.3** | | 1. Browse and load “Broken Model 3.txt”. | A message box should pop up informing that “Transition is not in the valid format”. | **T1.4** | | 1. Browse and load “Broken Model 4.txt”. | A message box should pop up informing that “Transition t1 is not in [from state] – [to state] format”. | **T1.5** | | 1. Browse and load “Broken Model 5.txt”. | A message box should pop up informing that “Transition from state s1 to state s2 are defined more than once”. | **T1.6** | | 1. Browse and load “Broken Model 6.txt”. | A message box should pop up informing that “Invalid state is detected in transition t1”. | **T1.7** | | 1. Browse and load “Broken Model 7.txt”. | A message box should pop up informing that “Atom p is defined more than once for state s1”. | **T1.8** | | 1. Browse and load “Broken Model 8.txt”. | A message box should pop up informing that “State s5 is not defined”. | **T1.9** | | |  | | --- | | **Result** | | Pass 🗹Fail 🞎N/A 🞎 | | **Initials/Date** | | 2020 | |
| T2 | |  |  |  | | --- | --- | --- | | **This test demonstrates that given the proper model input kripke structure file, valid State ID and CTL formula, the system checks the formula for the given state ID and return the correct result as expected.**  **Each test formula is formatted as follow:** [state ID];[CTL Formula];[Expected Result]  **For example:**   * s1;EXq;True **means that state s1 holds for formula EXq** * s4;EGp and AGq;False **means that state s4 does NOT hold for formula EGp and AGq** | | | | 1. Open ModelCheckCTL.Test. | -- | -- | | 1. Print Model 1.txt | s1, s2, s3, s4;  t1 : s1 - s2,  t2 : s1 - s3,  t3 : s3 - s4,  t4 : s4 - s2,  t5 : s2 - s3;  s1 : p q,  s2 : q t r,  s3 : ,  s4 : t; | T2.3 | | 1. Print Model 1 – Test Formulas.txt | s1;EGp and AGq;False  s2;EGp and AGq;False  s3;EGp and AGq;False  s4;EGp and AGq;False  s1;(not AGp) or EFq;True  s2;(not AGp) or EFq;True  s3;(not AGp) or EFq;True  s4;(not AGp) or EFq;True  s1;EG(r->t);True  s2;EG(r->t);True  s3;EG(r->t);True  s4;EG(r->t);True  s1;AX(r->p);False  s2;AX(r->p);True  s3;AX(r->p);True  **s4;AX(r->p);False**  s1;AXq;False  s2;AXq;False  s3;AXq;False  s4;AXq;True  s1;EXq;True  s2;EXq;False  s3;EXq;False  s4;EXq;True  s1;not AXq;True  s2;not AXq;True  s3;not AXq;True  s4;not AXq;False  s1;not EXq;False  s2;not EXq;True  s3;not EXq;True  s4;not EXq;False  s1;A(pUq);True  s2;A(pUq);True  s3;A(pUq);False  s4;A(pUq);False  s1;E(pUq);True  s2;E(pUq);True  s3;E(pUq);False  s4;E(pUq);False  s1;AXq and A(pUq);False  s2;AXq and A(pUq);False  s3;AXq and A(pUq);False  s4;AXq and A(pUq);False  s1;AXq or A(pUq);True  s2;AXq or A(pUq);True  s3;AXq or A(pUq);False  s4;AXq or A(pUq);True  s1;EFr;True  s2;EFr;True  s3;EFr;True  s4;EFr;True  s1;AFr;True  s2;AFr;True  s3;AFr;True  s4;AFr;True  s1;EGt;False  s2;EGt;False  s3;EGt;False  s4;EGt;False  s1;AGq;False  s2;AGq;False  s3;AGq;False  s4;AGq;False  s1;AX((EFp)or(AFr));True  s2;AX((EFp)or(AFr));True  s3;AX((EFp)or(AFr));True  s4;AX((EFp)or(AFr));True  s1;EX((AFp)or(EFr));True  s2;EX((AFp)or(EFr));True  s3;EX((AFp)or(EFr));True  s4;EX((AFp)or(EFr));True  s1;A(pUA(qUr));False  s2;A(pUA(qUr));True  s3;A(pUA(qUr));False  s4;A(pUA(qUr));False  s1;E(A(qUr)Ut);False  s2;E(A(qUr)Ut);True  s3;E(A(qUr)Ut);False  s4;E(A(qUr)Ut);True  s1;AG(p->A(pU(not p and A(not pUq))));True  s2;AG(p->A(pU(not p and A(not pUq))));True  s3;AG(p->A(pU(not p and A(not pUq))));True  s4;AG(p->A(pU(not p and A(not pUq))));True | **T2.3** | | 1. Print Model 2.txt | s0, s1, s2, s3;  t1 : s0 - s1,  t2 : s0 - s3,  t3 : s1 - s1,  t4 : s1 - s2,  t5 : s2 - s0,  t6 : s2 - s3,  t7 : s3 - s0;  s0 : p q,  s1 : r,  s2 : p t,  s3 : q r; | **T2.5** | | 1. Print Model 2 – Test Formulas.txt | s0;AFq;True  s1;AFq;False  s2;AFq;True  s3;AFq;True  s0;AG(EF(p or r));True  s1;AG(EF(p or r));True  s2;AG(EF(p or r));True  **s3;AG(EF(p or r));True**  s0;EXr;True  s1;EXr;True  s2;EXr;True  s3;EXr;False  s0;EX(EXr);True  s1;EX(EXr);True  s2;EX(EXr);True  s3;EX(EXr);True  s0;AXr;True  s1;AXr;False  s2;AXr;False  s3;AXr;False  s0;AG(AFq);False  s1;AG(AFq);False  s2;AG(AFq);False  s3;AG(AFq);False | **T2.5** | | 1. Print Model 3.txt | s1, s2, s3, s4, s5, s6, s7, s8, s9, s10;  t1 : s1 - s2,  t2 : s2 - s3,  t3 : s3 - s4,  t4 : s4 - s3,  t5 : s2 - s5,  t6 : s5 - s6,  t7 : s6 - s7,  t8 : s7 - s6,  t9 : s5 - s8,  t10 : s8 - s10,  t11 : s9 - s10,  t12 : s10 - s9,  t13 : s9 - s8;  s1 : ,  s2 : p,  s3 : p q,  s4 : q,  s5 : p,  s6 : p q,  s7 : p,  s8 : p q,  s9 : p q,  s10 : p q; | T2.7 | | 1. Print Model 3 – Test Formulas.txt | s1;AGp;False  s2;AGp;False  s3;AGp;False  s4;AGp;False  s5;AGp;True  s6;AGp;True  s7;AGp;True  s8;AGp;True  s9;AGp;True  s10;AGp;True  s1;EGp;False  s2;EGp;True  s3;EGp;False  s4;EGp;False  **s5;EGp;True**  s6;EGp;True  s7;EGp;True  s8;EGp;True  s9;EGp;True  s10;EGp;True  s1;AGq;False  s2;AGq;False  s3;AGq;True  s4;AGq;True  s5;AGq;False  s6;AGq;False  s7;AGq;False  s8;AGq;True  s9;AGq;True  s10;AGq;True  s1;EGq;False  s2;EGq;False  s3;EGq;True  s4;EGq;True  s5;EGq;False  s6;EGq;False  s7;EGq;False  s8;EGq;True  s9;EGq;True  s10;EGq;True | **T2.7** | | 1. Print Model 4.txt | s1, s2, s3, s4, s5, s6, s7, s8, s9;  t1 : s1 - s2,  t2 : s2 - s3,  t3 : s3 - s4,  t4 : s5 - s4,  t5 : s2 - s5,  t6 : s4 - s6,  t7 : s8 - s2,  t8 : s1 - s6,  t9 : s6 - s7,  t10 : s7 - s8,  t11 : s9 - s8,  t12 : s6 - s9;  s1 : n1 n2 0,  s2 : t1 n2 1,  s3 : c1 n2 1,  s4 : c1 t2 1,  s5 : t1 t2 1,  s6 : n1 t2 2,  s7 : t1 t2 2,  s8 : t1 c2 2,  s9 : n1 c2 2; | T2.8 | | 1. Print Model 4 – Test Formulas.txt | s1;AG(t1 -> AF c1);True  s2;AG(t1 -> AF c1);True  s3;AG(t1 -> AF c1);True  **s4;AG(t1 -> AF c1);True**  s5;AG(t1 -> AF c1);True  s6;AG(t1 -> AF c1);True  s7;AG(t1 -> AF c1);True  s8;AG(t1 -> AF c1);True  s9;AG(t1 -> AF c1);True | **T2.9** | | 1. Print Model 5.txt | s1, s2, s3, s4, s5, s6, s7;  t1 : s1 - s2,  t2 : s1 - s3,  t3 : s2 - s4,  t4 : s2 - s5,  t5 : s3 - s6,  t6 : s3 - s7;  s1 : ,  s2 : p,  s3 : p,  s4 : p,  s5 : ,  s6 : ,  s7 : ; | **T2.11.1** | | 1. Print Model 5 – Test Formulas.txt | **s1;AXp;True**  s2;AXp;False  s3;AXp;False  s4;AXp;False  s5;AXp;False  s6;AXp;False  s7;AXp;False  s1;EXp;True  s2;EXp;True  s3;EXp;False  s4;EXp;False  s5;EXp;False  s6;EXp;False  s7;EXp;False | **T2.11.1**  **T2.11.2** | | 1. Print Model 6.txt | s1, s2, s3, s4, s5, s6, s7;  t1 : s1 - s2,  t2 : s1 - s3,  t3 : s2 - s4,  t4 : s2 - s5,  t5 : s3 - s6,  t6 : s3 - s7;  s1 : ,  s2 : ,  s3 : ,  s4 : p,  s5 : p,  s6 : p,  s7 : ; | **T2.13** | | 1. Print Model 6 – Test Formulas.txt | s1;AFp;False  **s2;AFp;True**  s3;AFp;False  s4;AFp;True  s5;AFp;True  s6;AFp;True  s7;AFp;False  s1;EFp;True  s2;EFp;True  s3;EFp;True  s4;EFp;True  s5;EFp;True  s6;EFp;True  s7;EFp;False | **T2.13** | | 1. Print Model 7.txt | s1, s2, s3, s4, s5, s6, s7;  t1 : s1 - s2,  t2 : s1 - s3,  t3 : s2 - s4,  t4 : s2 - s5,  t5 : s3 - s6,  t6 : s3 - s7;  s1 : p,  s2 : p,  s3 : p,  s4 : p q,  s5 : ,  s6 : q,  s7 : q; | **T2.15.1** | | 1. Print Model 7 – Test Formulas.txt | **s1;A(p U q);False**  s2;A(p U q);False  s3;A(p U q);True  s4;A(p U q);True  s5;A(p U q);False  s6;A(p U q);True  s7;A(p U q);True  s1;E(p U q);True  s2;E(p U q);True  s3;E(p U q);True  s4;E(p U q);True  s5;E(p U q);False  s6;E(p U q);True  s7;E(p U q);True | **T2.15.1**  **T2.15.2** | | 1. Run all test formulas with their expected results as described above. | * All formulas matched their expected results. * All test results are marked as passed | T2.16 | | |  | | --- | | **Result** | | Pass 🗹Fail 🞎N/A 🞎 | | **Initials/Date** | | GS/2020 | |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Attachments |  |
| Microwave\_oven.txt | s1, s2, s3, s4, s5, s6, s7;  t1 : s1 - s2, t2 : s2 - s5, t3 : s5 - s2, t4 : s5 - s3, t5 : s1 - s3, t6 : s3 - s1, t7 : s3 - s6, t8 : s1 - s4, t9 : s4 - s4, t10: s4 - s3, t11: s7 - s4, t12: s6 - s7;  s1:, s2: s e, s3: c, s4: c h, s5: s c e, s6: s c, s7: s c h; | **T.w.1.1** | |  | | --- | | **Result** | | Pass 🗹Fail 🞎N/A 🞎 | | **Initials/Date** | | GS/2020 | |
| Test formulas | **S2;EG (not h);True**  **S2;not( EF(s and EG (not h)));False** | **T.w.1.1**  **T.w.1.2**  **T.w.2.1**  **T.w.2.2** |  |