Structural Similarity of Boundary Conditions and an Efficient Local Search Algorithm for Goal Conflict Identification

Hongzhen Zhong

Sun Yat-sen University, China
The 27^{th} Asia-Pacific Software Engineering Conference

1-4 December 2020 – Singapore

The Background of Goal Conflict Identification

- In goal-oriented requirements engineering, requirements are described as domain properties and goals
- Domain properties are descriptive statements that capture the domain about the problem world
- Goals are prescriptive statements that the system must achieve
- Domain properties and goals are represented in the Linear-time Temporal Logic (LTL).
- Goal conflict identification aims to find inconsistencies between goals.
- Boundary conditions (BCs) capture such inconsistencies under which goals are unsatisfiable as a whole.
- BC is also represented in LTL.

Definition of Goal Conflict Identification

Definition (Goal Conflict Identification)

Input: domain properties and goals.
Output: a set of boundary conditions.

Example (MinePump)

Minepump: consider a system to control a pump inside a mine. The main goal of the system is to avoid flood in the mine. The system has two sensors.

- One detects the high water level (h)
- The other detects methane in the environment (m)
- When the water level is high, the system should turn on the pump (p)
- When there is methane in the environment, the pump (p) should be turned off.

Example (MinePump)

Domain Property:

1. **Name**: PumpEffect
   Description: The pump is turned on for two time steps, then in the following one the water level is not high.
   Formula: $\Box((p \wedge \bigcirc p) \rightarrow \bigcirc(\bigcirc \neg h))$

Goals:

1. **Name**: NoFlooding
   Description: When the water level is high, the system should turn on the pump.
   Formula: $\Box(h \rightarrow \bigcirc(p))$
2. **Name**: NoExplosion
   Description: When there is methane in the environment, the pump should be turned off.
   Formula: $\Box(m \rightarrow \bigcirc(\neg p))$

$$bc_1\colon \Diamond(h \wedge m)$$
$$bc_2\colon \Box(h \wedge m)$$

Definition of Boundary Condition

- *Dom* be domain properties
- $\{G_1, \ldots, G_n\}$ be a set of goals

φ is a *boundary condition*, such that the following properties hold:

$$\{Dom, \varphi, \bigwedge_{1 \leq i \leq n} G_i\} \models \bot \qquad \text{(logical inconsistency)}$$

$$\{Dom, \varphi, \bigwedge_{j \neq i} G_j\} \not\models \bot, \text{ for each } 1 \leq i \leq n \qquad \text{(minimality)}$$

$$\varphi \neq \neg(G_1 \wedge \ldots \wedge G_n) \qquad \text{(non-triviality)}$$

Definition of Boundary Condition

- *Dom* be domain properties
- $\{G_1, \ldots, G_n\}$ be a set of goals

φ is a *boundary condition*, such that the following properties hold:

$$\{Dom, \varphi, \bigwedge_{1 \leq i \leq n} G_i\} \models \bot \qquad \text{(logical inconsistency)}$$

$$\{Dom, \varphi, \bigwedge_{j \neq i} G_j\} \not\models \bot, \text{ for each } 1 \leq i \leq n \qquad \text{(minimality)}$$

$$\varphi \neq \neg(G_1 \wedge \ldots \wedge G_n) \qquad \text{(non-triviality)}$$

The BC captures a situation where the goals **as a whole** are not satisfiable.

Definition of General Boundary Condition

Definition (General Boundary Condition)

Given two boundary conditions bc_1 and bc_2, we call bc_1 is more general (or weaker) than bc_2 if bc_2 implies bc_1.

- in MinePump

$$bc_1 : \Diamond(h \wedge m)$$

$$bc_2 : \Box(h \wedge m)$$

- bc_1 is more general than bc_2 because $bc_2 \models bc_1$
- the more general BC is of higher quality
- once it is worked out, less general BCs will also be worked out.

The Challenge of Goal Conflict Identification

Challenges

1. How to identify the boundary conditions in time.
2. How to identify the more general boundary conditions.

Structural Similarity of Boundary Conditions

- We find an interesting phenomenon that there exist some pairs of BCs which are similar on the structure.

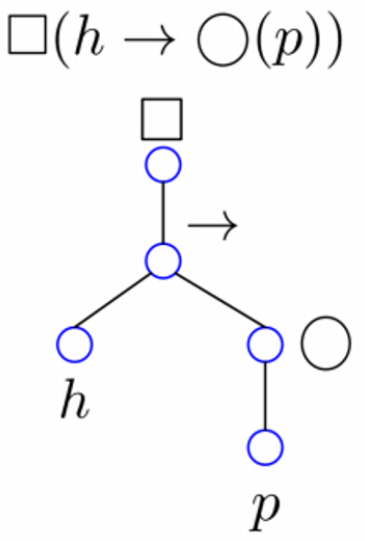
$$bc_1 : \Diamond(h \wedge m)$$

$$bc_2 : \Box(h \wedge m)$$

- Once one BC is found, the other one may be found via simply changing the former.

How frequently similar BC pairs do occur in cases?

- Think of BCs as syntax trees.



- Use tree edit distance (TED) [K. Zhang and D. Shasha, 1989] to measure the structural similarity of the BCs.
  - $\delta(\varphi, \varphi')$ is TED of φ and φ'.
  - $\Delta(\varphi, \varphi')$ is normalized TED of φ and φ. i.e., $\frac{\delta(\varphi,\varphi')}{|\varphi|+|\varphi'|}$.
  - $\%BC(\delta \leq l) = \frac{\#\{BC|\exists BC'.\delta(BC,BC')\leq l\}}{\#\text{total BC}}$.
  - $\#sim(\delta \leq l) = \frac{\sum\{BC'|\delta(BC,BC')\leq l\}}{\#\text{total BC}}$.
- Measure the BCs that the state-of-the-art approaches found.

How frequently similar BC pairs do occur in cases?

We generated BCs by applying the Tableaux [R. Degiovanni, 2016] and GA-based approache [R. Degiovanni, 2018] for 24 hours (10 runs in parallel).

Case	GA-based Approach and Tableaux													
	$\%BC(\delta \leq l)$			$\#\text{sim}(\delta \leq l)$			$\%BC(\Delta \leq k)$			$\#\text{sim}(\Delta \leq k)$			avg min	#total BC
	$l=1$	$l=2$	$l=3$	$l=1$	$l=2$	$l=3$	$k=0.1$	$k=0.2$	$k=0.3$	$k=0.1$	$k=0.2$	$k=0.3$		
RP1	85.3%	96.4%	98.0%	2.1	6.8	13.4	91.9%	97.1%	98.7%	5.1	21.0	42.3	1.1	308
RP2	84.2%	96.6%	98.8%	2.1	6.8	13.5	87.6%	97.2%	99.1%	8.1	39.4	61.0	1.1	566
Ele	65.9%	93.5%	97.6%	1.2	4.2	9.3	35.8%	90.2%	97.6%	1.0	4.8	11.6	1.2	124
TCP	69.3%	91.9%	98.7%	1.5	5.0	11.3	52.8%	93.2%	98.2%	1.7	6.9	16.2	1.1	382
AAP	63.2%	92.0%	97.9%	1.4	5.0	11.7	24.7%	85.8%	95.1%	0.5	3.4	7.3	1.2	289
MP	68.4%	93.4%	98.2%	1.5	5.4	12.7	54.2%	90.8%	96.8%	1.9	10.0	22.0	1.1	913
ATM	72.6%	94.6%	98.5%	1.7	5.8	12.0	77.9%	97.2%	98.5%	3.6	17.5	37.2	1.1	681
RRCS	40.8%	81.0%	90.2%	0.7	2.3	4.3	66.5%	93.4%	98.1%	1.6	6.6	14.7	1.5	317
Tel	39.2%	72.0%	82.9%	0.7	1.9	3.7	80.5%	95.1%	97.9%	3.1	10.5	23.9	1.7	534
LAS	80.4%	92.5%	99.1%	2.0	6.0	8.9	95.3%	97.2%	97.2%	18.0	24.4	27.4	1.1	108
PA	58.8%	79.1%	87.8%	1.1	3.0	5.6	74.7%	86.9%	92.8%	2.5	7.3	15.5	1.5	321
RRA	86.2%	95.6%	97.8%	2.1	7.0	15.1	92.5%	98.1%	99.4%	4.0	16.8	36.6	1.1	319
SA	29.4%	64.7%	76.5%	0.5	1.1	1.5	64.7%	82.4%	88.2%	2.1	3.8	7.1	1.8	18
LB	7.1%	29.8%	41.7%	0.2	0.5	0.7	63.1%	84.5%	91.7%	0.8	2.1	3.1	3.6	85
LC	0.9%	12.4%	27.8%	0.1	0.3	0.6	36.9%	66.5%	81.7%	0.9	3.4	6.8	4.2	443
AMBA	1.8%	43.3%	72.0%	0.1	1.1	2.0	38.4%	76.8%	91.5%	1.1	6.9	12.5	1.9	165

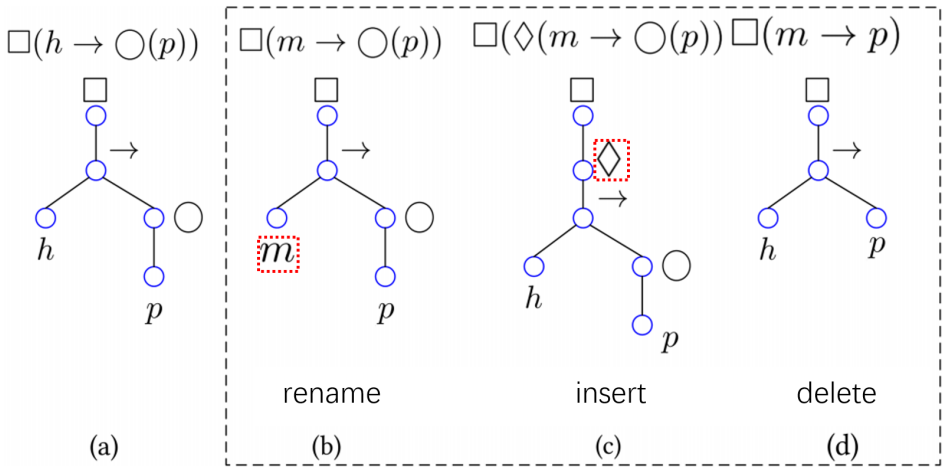
In most cases,

- more than 70% of the BCs have more than one ($\#sim(\delta \leq 3)$) BC nearby ($\%BC(\delta \leq 3)$);
- the results using normalized TED ($\%BC(\Delta \leq 3)$ and $\#sim(\Delta \leq 3)$) are similar to the first point.
- the average minimum distance (avg min) between BCs and their most similar BCs is less than 2.

Boundary Condition Identification – LOGION

We design a local search algorithm, LOGION, which uses the structural similarity of BCs to identify BCs.

- Inspire tree edit operations, design three LTL formula edit operations to slightly modify an LTL formula to another LTL formula.
  - Rename a symbol.
  - Insert an operator.
  - Delete an operator.
- The neighborhood function w.r.t. φ, denoted by $N(\varphi)$.
  all formulae obtained by applying a formula editing operation to a sub-formulae of φ.



Boundary Condition Identification – LOGION

- Initialization
  - the initial formula should be "close" to a BC and easy to construct.
  - $\neg(G_1 \wedge \cdots \wedge G_n)$.
- Objective Function
  - use the objective function proposed by [R. Degiovanni, 2018].
  - $f(\varphi) = \mathrm{li}(\varphi) + \sum_{i=1}^{|G|} \min(\varphi, G_{-i}) + \mathrm{nt}(\varphi) + \frac{1}{|\varphi|}$
    - $li(\varphi) = \begin{cases} 1 & \text{if } (Dom \wedge G \wedge \varphi) \text{ is UNSAT} \\ 0 & \text{otherwise} \end{cases}$
    - $min(\varphi, G_{-i}) = \begin{cases} \frac{1}{|G|} & \text{if } (Dom \wedge G_{-i} \wedge \varphi) \text{ is SAT} \\ 0 & \text{otherwise} \end{cases}$
    - $nt(\varphi) = \begin{cases} 0.5 & \text{if } \neg G \not\equiv \varphi \\ 0 & \text{otherwise} \end{cases}$
- The algorithm of LOGION
  - $\varphi \leftarrow$ initialization;
  - while the cutoff time is not reached do
    - $\bar{\mathfrak{B}} \leftarrow N(\varphi)$;
    - $\varphi* \leftarrow$ the formula in $\bar{\mathfrak{B}}$ with highest score;
    - $\mathfrak{B} \leftarrow \mathfrak{B} \cup$ BCs in $\bar{\mathfrak{B}}$;
  - return $\mathfrak{B}$;

To what extent does LOGION exploit the structural similarity of BCs?

Case	LOGION	
	$\delta(BC_i, BC_{i+1})$	$\Delta(BC_i, BC_{i+1})$
RP1	6.43	0.18
RP2	5.70	0.21
Ele	4.72	0.25
TCP	5.24	0.23
AAP	5.32	0.31
MP	4.69	0.21
ATM	5.79	0.22
RRCS	5.70	0.19
Tel	5.39	0.24
LAS	5.75	0.09
PA	4.36	0.09
RRA	5.36	0.14
SA	5.70	0.10
LB	10.43	0.05
LC	17.72	0.10
AMBA	24.67	0.09

- "$\delta(BC_i, BC_{i+1})$": the average distance between BC_i and its next BC_{i+1} computed.
- "$\Delta(BC_i, BC_{i+1})$" means the average normalized distance between BC_i and its next BC_{i+1} computed.

How does LOGION compare against the state-of-the-art competitors on cases?

Cutoff time: 1 hour.

Case	Tableaux				GA-based Approach						LOGION					
	#BC	#gen.	rank	Time	#BC	#gen.	rank	T_{FBC}	S_{BBC}	#suc.	#BC	#gen.	rank	T_{FBC}	S_{BBC}	#suc.
RP1	1.0	0.9	2.9	0.1	45.2	2.8	**1.4**	8.8	14.2	**10**	**6935.5**	**4.0**	1.5	**3.5**	**8.8**	**10**
RP2	1.0	0.7	1.6	0.4	48.0	3.0	1.4	18.7	16.8	**10**	**14158.0**	**7.0**	**1.3**	**1.4**	**7.6**	**10**
Ele	-	-	-	-	41.2	2.1	**1.5**	4.1	**5.0**	**10**	**7759.4**	**22.2**	**1.5**	**1.8**	**5.0**	**10**
TCP	2.0	0.6	2.2	1.0	80.4	0.4	2.0	17.2	**5.1**	**10**	**10335.4**	**20.0**	**1.0**	**0.9**	7.3	**10**
AAP	4.0	4.0	2.3	0.3	56.8	1.1	2.2	25.1	**3.0**	**10**	**22512.9**	**41.9**	**1.1**	**4.4**	**3.0**	**10**
MP	-	-	-	-	75.2	3.0	1.9	10.0	**3.0**	**10**	**27112.2**	**26.9**	**1.1**	**0.3**	**3.0**	**10**
ATM	3.0	2.0	2.4	1.9	98.6	1.4	1.9	12.0	**6.0**	**10**	**8773.6**	**88.3**	**1.4**	**0.6**	12.4	**10**
RRCS	-	-	-	-	60.4	4.2	1.6	7.5	11.0	**10**	**17912.0**	**113.4**	**1.2**	**1.1**	**6.4**	**10**
Tel	-	-	-	-	155.0	4.2	2.0	61.3	12.7	9	**12408.2**	**89.1**	**1.0**	**1.0**	**6.6**	**10**
LAS	-	-	-	-	36.8	0.9	**1.2**	913.3	43.0	6	**12125.4**	**133.3**	1.3	**1.2**	**20.6**	**10**
PA	-	-	-	-	-	-	-	-	-	0	**3220.6**	**45.9**	**1.0**	**2.4**	**15.4**	**10**
RRA	-	-	-	-	88.9	0.4	2.0	470.1	12.6	**10**	**1966.5**	**80.2**	**1.0**	**1.2**	**10.9**	**10**
SA	-	-	-	-	-	-	-	-	-	0	**3600.8**	**42.9**	**1.0**	**11.3**	**20.7**	**10**
LB	-	-	-	-	-	-	-	-	-	0	**87.8**	**2.8**	**1.0**	**1489.2**	**62.5**	**4**
LC	-	-	-	-	-	-	-	-	-	0	**16.6**	**1.7**	**1.0**	**389.3**	**91.6**	**7**
AMBA	-	-	-	-	-	-	-	-	-	0	-	-	-	-	-	0

- "#BC": the average number of BCs in 10 runs.
- "#gen.": the average number of general BCs in 10 runs.
- "rank": the average ranking of the more general BCs with the highest likelihood.
- "T_{FBC}": the time (second) of identifying the first BC.
- "S_{BBC}": the size of the best BC (the most compact BC).
- "#suc.": the number of successful runs (out of 10 runs).

Conclusion and Future Work

Conclusion

- discover an interesting phenomenon that there are some pairs of BCs are similar in formula structure.
- using this phenomenon, we propose an efficient local search algorithm named LOGION for goal-conflict identification.

Future Work

- optimize the BC verification procedure by reducing the calls of the LTL satisfiability checker.

Thank

Thank you