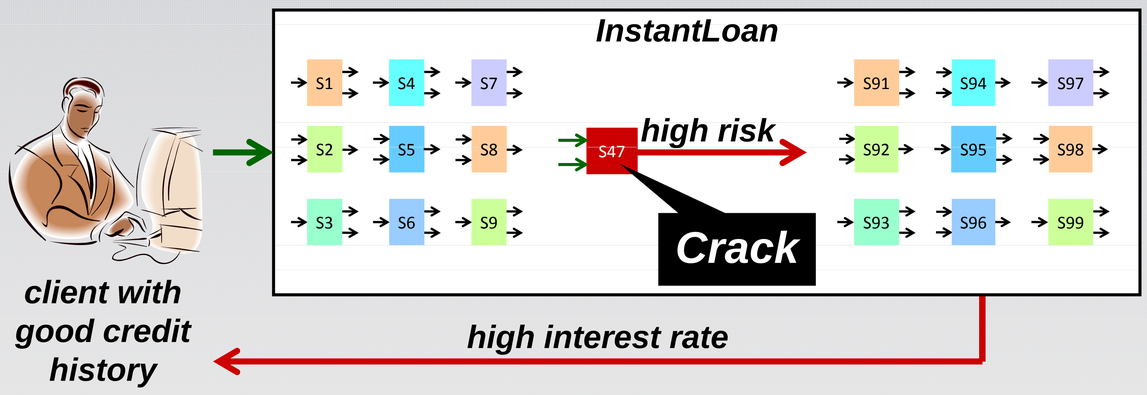
Where to Adapt Dynamic Service Compositions

Bo Jiang[1], W. K. Chan[2], Zhenyu Zhang[1], and T. H. Tse[1]

[1]*The University of Hong Kong* [2]*City University of Hong Kong*

Motivating Example



Crack

- A crack is a low-quality service
- Real-life service compositions are large in scale, deep in service invocation chains, data-dependent on dynamic behaviors of member services, and complex to analyze in detail
- Cracks are hidden behind many other services and hard to find
- Our task is to find cracks and their service endpoints automatically with a view to adapting alternative candidate services

Service Monitoring and QoS Collection

Whenever a service endpoint invokes another service, sample:

- the invoked service
- QoS values of the invoker.

Collect a value vector $v = \langle v_1, v_2, \ldots, v_m \rangle$ representing the set of given QoS metrics values at the endpoint.

Compute

- $N(v)$ = total number of service invocations in the invocation history
- $N(v, s)$ = number of innovations for service s
- $\overline{v(s)}$ = mean value of all the elements v_i in v related to s.

Crack-Sensitivity

Crack-sensitivity of a service s is given by the equation

$$M(s) = \frac{\sum_{v \in V} \frac{N(v,s)}{N(v)} \left(1 - \overline{v(s)}\right)}{\sum_{v \in V} \frac{N(v,s)}{N(v)}}$$

Ranks a service higher if:

- More often selected for execution by peer services
- The QoS according to the service vectors at invocation endpoints are low.

Identifying Cracks

- Consider ranking history of multiple executions
- Services with consistently high crack-sensitivities are deemed as potential cracks
- Use analysis of variance to determine consistency

Consistent ranking ***⇒ crack***

Inconsistent ranking ***⇒ not crack***

Experimentation

Goal: To verify the effectiveness of our technique

Case: Motor vehicle insurance case study in [1, 2]. Five services: Europe Assist service (EA), AGFIL service, Garage service (G), Lee Consulting Services (Lee), and Loss Adjustor service (LA)

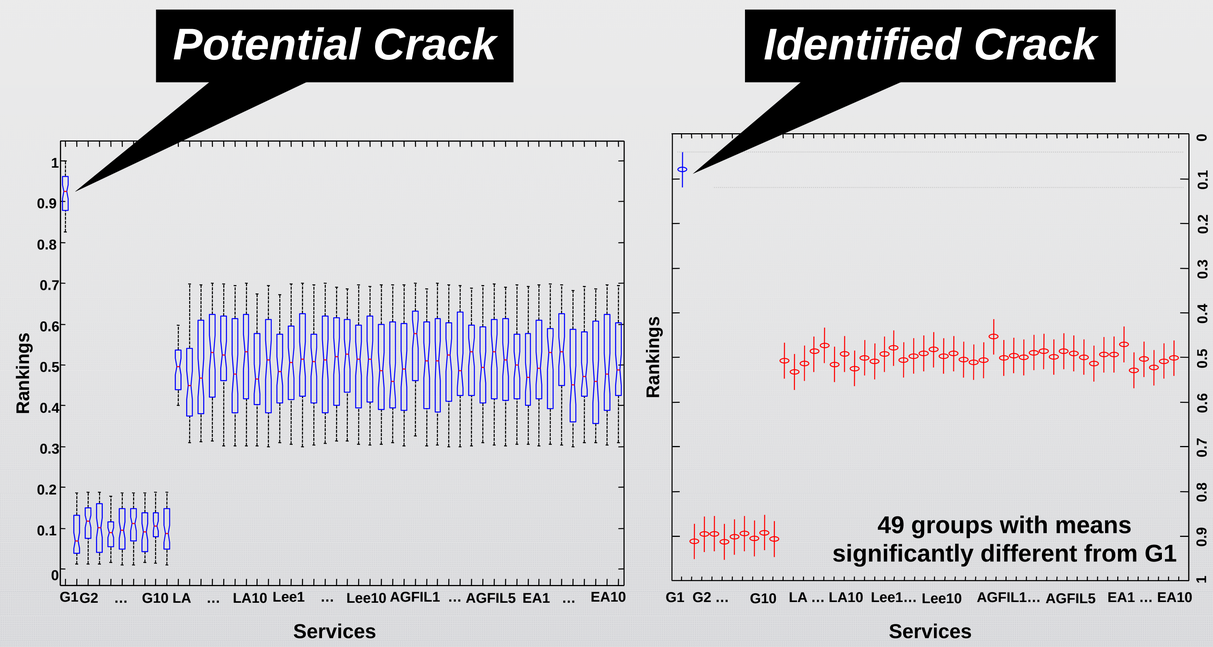
***Setup*:**

- 10 instances of each kind of services in service pool for selection
- Assign one garage (G1) to exhibit 1% chance of malfunction
- Compose services according to [2]

***Procedure*:**

- Invoke composite service 100,000 times
- Malfunction of G1 leads to abortion of composite service
- Rank service instances at each invocation of EA
- Analysis of variance to determine consistency of ranking history
- Multiple-comparison to determine the crack service

***Results*:** Based on G1, the service endpoint "Assign garage" in EA is reported as crack



Analysis of variance of service crack-sensitivity

Confirming the crack by multiple comparisons

Selected References

[1] CrossFlow Consortium/AGFIL. Insurance requirements. *CrossFlow deliverable D1.b*. La Gaude, March 1999.

[2] C. Ye, S.C. Cheung, W.K. Chan, and C. Xu. Atomicity analysis of service composition across organizations. *IEEE Transactions on Software Engineering* 35 (1): 2–28, 2009.