Context transformers in K framework

Andrei Arusoaie and Traian Florin Șerbănuță

Context Transformers

- K rule:

```
rule [Variable lookup] :
<k open="right">(X => V)</k>
<env open="both"> X |-> L</env>
<store open="both"> L |-> V</store>
```

- we only specify cells k, env and store
- we don't have to change the rule if we add more cells in the configuration

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    ...
</T>
```

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    ...
</T>
```

```
rule [Variable lookup] :
    <k open="right">(X => V)</k>
    <env open="both"> X |-> L</env>
    <store open="both"> L |-> V</store>
```

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    ...
</T>
```

```
rule [Variable lookup] :
    <k open="right">(X => V)</k>
    <env open="both"> X |-> L</env>
    <store open="both"> L |-> V</store>
```

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    ...
</T>
```

```
rule [Variable lookup] :
    <k open="right">(X => V)</k>
    <env open="both"> X |-> L</env>
    <store open="both"> L |-> V</store>
```

```
rule [Variable lookup] :
    <server>
        <threads>
            <thread>
                <k open="right">(X => V)</k>
                <env open="both"> X |-> L</env>
            ...</thread>
        </threads>
        <store open="both"> L |-> V</store>
    </server>
```

Context transformers in K 2.0

- do not allow us to use the same label for two different cells
  - the cases when context transformers could have more than one solution are not analyzed
- only minimal context is computed
- rule generation ?

Context transformers in K 3.0

- we can use the same label for two different cells
  - Problem: ambiguities
  - Solution: generate all possibilities and then apply:
    - consistency checks
    - locality principle
- compute both full and minimal context

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    <client color="red" multiplicity="*">
        <k color="red">.K</k>
        <env color="red">.Map</env>
        <store color="red">.Map</store>
        <bar color="red">.Map</bar>
    </client>
</T>
```

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    <client color="red" multiplicity="*">
        <k color="red">.K</k>
        <env color="red">.Map</env>
        <store color="red">.Map</store>
        <bar color="red">.Map</bar>
    </client>
</T>
```

```
rule [Variable lookup] :
    <k open="right">(X => V)</k>
    <env open="both"> X |-> L</env>
    <store open="both"> L |-> V</store>
```

- Step 1: Find all mappings

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    <client color="red" multiplicity="*">
        <k color="red">.K</k>
        <env color="red">.Map</env>
        <store color="red">.Map</store>
        <bar color="red">.Map</bar>
    </client>
</T>
```

```
rule [Variable lookup] :
    <k open="right">(X => V)</k>
    <env open="both"> X |-> L</env>
    <store open="both"> L |-> V</store>
```

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    <client color="red" multiplicity="*">
        <k color="red">.K</k>
        <env color="red">.Map</env>
        <store color="red">.Map</store>
        <bar color="red">.Map</bar>
    </client>
</T>
```

```
rule [Variable lookup] :
    <k open="right">(X => V)</k>
    <env open="both"> X |-> L</env>
    <store open="both"> L |-> V</store>
```

Example

```
configuration
<T color="red">
    <server color="red" multiplicity="*">
        <threads color="red">
            <thread color="red" multiplicity="*">
                <k color="red">.K</k>
                <env color="red">.Map</env>
                <baz color="red">.Map</baz>
            </thread>
        </threads>
        <store color="red">.Map</store>
    </server>
    <client color="red" multiplicity="*">
        <k color="red">.K</k>
        <env color="red">.Map</env>
        <store color="red">.Map</store>
        <bar color="red">.Map</bar>
    </client>
</T>
```

```
rule [Variable lookup] :
    <k open="right">(X => V)</k>
    <env open="both"> X |-> L</env>
    <store open="both"> L |-> V</store>
```

- Step 2: Generate all solutions
  - consistency checks vs. “locality principle”
- Step 3: Compute minimal context
- Step 4: Apply locality principle

Conclusions + Future work

- cells with same label
- full context + minimal context
- generate rules