fn make_sized_bounded_int_monotonic_sum

Bridget Ma

Proves soundness of make_sized_bounded_int_monotonic_sum.

make_sized_bounded_int_monotonic_sum returns a Transformation that computes the sum of bounded ints, where all values share the same sign.

1 Hoare Triple

Precondition

- T (atomic input type) is a type with trait Integer. Integer implies T has the trait bound:
  - CheckNull so that T is a valid atomic type for AtomDomain
  - CheckAtom to satisfy the preconditions of new_closed
  - Zero provides a way to retrieve T's representation of 0
  - InfCast for casting a dataset distance of type usize to T
  - InfSub so that the output domain is compatible with the output metric and for subtracting an upper bound of type T by a lower bound of type T

Pseudocode

```
def make_sized_bounded_int_checked_sum(
    size: usize,
    bounds: (T, T)
):
    input_domain: VectorDomain[AtomDomain[T].new_closed(bounds)].with_size(size), #
    input_metric: SymmetricDistance, #
    input_metric: SymmetricDistance, #
    output_domain = AtomDomain(T) #

    if not signs_agree(bounds): #
        raise MakeTransformation("monotonic summation requires bounds to share" \
        " the same sign")
    (lower, upper) = bounds.clone() #
    range = upper.inf_sub(lower) #

    def function(arg: Vec[T]) -> T: #
        return arg.iter().fold(T.zero(), sum, v sum.saturating_add(v)) #

    output_metric = AbsoluteDistance(T)

    def stability_map(d_in: IntDistance) -> AbsoluteDistance<T>:
        substitutions = T.inf_cast(d_in / 2) #
        return substitutions.inf_mul(range) #

    return Transformation(
        input_domain, output_domain, function,
        input_metric, output_metric, stability_map)
```

Postcondition

Theorem 1.1. For every setting of the input parameters `(size, bounds)` to `make_sized_bounded_int_monotonic_sum` such that the given preconditions hold, `make_sized_bounded_int_monotonic_sum` raises an exception (at compile time or run time) or returns a valid transformation. A valid transformation has the following properties:

1. (Appropriate output domain). For every element x in `input_domain`, `function`(x) is in `output_domain` or raises a data-independent runtime exception.

2. (Stability guarantee). For every pair of elements x, x' in `input_domain` and for every pair (`d_in`, `d_out`), where `d_in` has the associated type for `input_metric` and `d_out` has the associated type for `output_metric`, if x, x' are `d_in`-close under `input_metric`, `stability_map`(`d_in`) does not raise an exception, and `stability_map`(`d_in`) $\leq$ `d_out`, then `function`(x), `function`(x') are `d_out`-close under `output_metric`.

2 Proofs

Proof. **(Part 1 – appropriate output domain).** The `output_domain` is `AtomDomain(T)`, so it is sufficient to show that `function` always returns non-null values of type `T`. By the definition of the `SaturatingAdd` trait, `T.saturating_add` always returns a non-null value of type `T`. Thus, in all cases, the function (from line 16) returns a non-null value of type `T`. □

Before proceeding with proving the validity of the stability map, we provide a couple lemmas.

Lemma 2.1. Let i, j, k be integers. $|\min(i, k) - \min(j, k)| \leq |i - j|$.

Proof. W.L.O.G. assume $i \geq j$. We have the following three cases

- $j \leq i \leq k$:

$$|\min(i, k) - \min(j, k)| = |i - j| \qquad \text{by } j \leq i \leq k$$

- $k < j \leq i$:

$$\begin{aligned} |\min(i, k) - \min(j, k)| &= |k - k| && \text{by } k < j \leq i \\ &\leq |i - j| && \text{by } 0 \leq |i - j| \end{aligned}$$

- $j \leq k < i$:

$$\begin{aligned} |\min(i, k) - \min(j, k)| &= |k - j| && \text{by } j \leq k < i \\ &\leq |i - j| && \text{by } j \leq k < i \end{aligned}$$

The above three cases are exhaustive given $i \geq j$. □

Lemma 2.2. Let i, j, k be integers. $|\max(i, k) - \max(j, k)| \leq |i - j|$.

Proof. W.L.O.G. assume $i \geq j$. We have the following three cases

- $j \leq i \leq k$:

$$\begin{aligned} |\max(i, k) - \max(j, k)| &= |k - k| && \text{by } j \leq i \leq k \\ &\leq |i - j| && \text{by } 0 \leq |i - j| \end{aligned}$$

- $k < j \leq i$:

$$|\max(i,k) - \max(j,k)| = |i - j| \qquad \text{by } k < j \leq i$$

- $j \leq k < i$:

$$\begin{aligned} |\max(i,k) - \max(j,k)| &= |i - k| && \text{by } j \leq k < i \\ &\leq |i - j| && \text{by } j \leq k < i \end{aligned}$$

The above three cases are exhaustive given $i \geq j$. $\square$

Lemma 2.3. For vector u, v with each element $\ell \in u$, $z \in v$ drawn from domain $\mathcal{X}$ and $\texttt{len(u)} = \texttt{len(v)} = \texttt{size}$, denote U and V as the multisets of the elements u and v respectively. Let $A = U \cap V$. $|\texttt{function(u)} - \texttt{function(v)}| \leq |\sum_{z \in U \setminus A} z - \sum_{z \in V \setminus A} z|$.

Proof. We have that

$$\begin{aligned} |\sum_{z \in U} z - \sum_{z \in V} z| &= |(\sum_{z \in U \setminus A} z + \sum_{z \in A} z) - (\sum_{z \in V \setminus A} z + \sum_{z \in A} z)| && \text{by properties of sets} && (1) \\ &= |\sum_{z \in U \setminus A} z - \sum_{z \in V \setminus A} z| && \text{by algebra} && (2) \end{aligned}$$

By the definition of `SaturatingAdd`, the invocation of `T.saturating_add` on line 17 saturates the sum at the relevant high or low boundary of `T`. Therefore, $\texttt{function(u)} = \min(\sum_{i \in u} i, c)$ where $c = T$. Therefore, $\sum_{z \in U} z$ and $\sum_{z \in V} z$ is equivalent to `sum(v)` and `sum(u)` respectively.

$$|\sum_{z \in U} z - \sum_{z \in V} z| = |\texttt{sum(u)} - \texttt{sum(v)}| = |\sum_{z \in U \setminus A} z - \sum_{z \in V \setminus A} z|$$

$\square$

Lemma 2.4. For vector u, v with each element $\ell \in u$, $z \in v$ drawn from domain $\mathcal{X}$ and $\texttt{len(u)} = \texttt{len(v)} = \texttt{size}$, $|\texttt{function}(u) - \texttt{function}(v)| \leq \frac{d_{Sym}(u,v)}{2} \cdot \texttt{range}$

Proof. Using the same notation as in 2.3, given $\texttt{len(u)} = \texttt{len(v)} = \texttt{size}$, by the definition of `ChangeOneDistance`, $|U \setminus A| = |V \setminus A| = d_{CO}(u,v) = d_{Sym}(u,v)/2$.
By pseudocode line 17, $\texttt{function}(u) = \texttt{sum}(u)$. Let $\alpha = \max_{(x,y) \in (U \setminus A \times V \setminus A)|x-y|}$. Given that every $\ell \in u$ must satisfy $\texttt{lower} \leq \ell \leq \texttt{upper}$, $\alpha \leq \texttt{upper} - \texttt{lower} = \texttt{range}$, where $\texttt{upper} - \texttt{lower} = \texttt{range}$ by pseudocode line 14.

$$\begin{aligned} |\texttt{function}(u) - \texttt{function}(v)| &= |\texttt{sum(u)} - \texttt{sum(v)}| \\ &= |\sum_{z \in U \setminus A} z - \sum_{z \in V \setminus A} z| && \text{by } 2.3 \\ &\leq |U \setminus A| \cdot \alpha && \text{by algebra} \\ &= \frac{d_{Sym}(u,v)}{2} \cdot \alpha \\ &\leq \frac{d_{Sym}(u,v)}{2} \cdot \texttt{range} \end{aligned}$$

$\square$

Proof. **(Part 2 – stability map).** Take any two elements u, v in the `input_domain` and any pair (`d_in`, `d_out`), where `d_in` has the associated type for `input_metric` and `d_out` has the associated type for `output_metric`. Assume u, v are `d_in`-close under `input_metric` and that `stability_map`(`d_in`) $\leq$ `d_out`. These assumptions are used to establish the following inequality:

$$
\begin{aligned}
|\texttt{function}(u) - \texttt{function}(v)| &\leq \frac{d_{Sym}(u, v)}{2} \cdot \texttt{range} && \text{by 2.4} \\
&\leq \frac{\texttt{d\_in}}{2} \cdot \texttt{range} && \text{by } \texttt{InfCast} \\
&\leq \texttt{T.inf\_cast(d\_in/2).inf\_mul(range)} && \text{by } \texttt{InfMul} \\
&= \texttt{stability\_map(d\_in)} && \text{by pseudocode line 23} \\
&\leq \texttt{d\_out} && \text{by the second assumption}
\end{aligned}
$$

It is shown that `function`(u), `function`(v) are `d_out`-close under `output_metric`. □