# Extended Example on TENTRIS

In this example, we show how the SPARQL query in Listing 1.1 is applied to the RDF graph from Listing 1.2 using using tensors and tensor operations. Further, we show how the hypertrie encodes the tensors and supports the operations used.

## 1 Tensor Representation of the RDF Graph

Consider the RDF Graph $g$ from Listing 1.1. A possible $id$ function that maps RDF terms to IDs is given in table Table 1. Note that there is an additional ID for unbound variables in solution mappings as we use the same $id$ function for the result. In table 2 the translation of the RDF terms from Listing 1.2 to tensor keys is provided. In the tensor representation $T := t(g)$ of $g$, the values for those keys are set to 1. All other entries are set to 0. A visualization of $t(g)$ is shown in Figure 1. Each 1 entry is shown as a dot.

## 2 SPARQL to Tensor Operations

A SPARQL query is translated to tensor operations in two steps: First, each triple pattern (TP) is transformed into a slice of the tensor $T$ with respect to $id$. To obtain the slice key from a triple pattern, terms are replaced with their IDs and variables are replaced with slice placeholder :.

The slices of $T$, which are equivalent to applying the triple patterns $\text{TP}^{(1)}$, $\text{TP}^{(2)}$ and $\text{TP}^{(3)}$ from Listing 1.1 to $g$, are:

$$\begin{aligned}\text{TP}^{(1)}(g) &\equiv T^{(1)} := T[1, 2, :] \\ \text{TP}^{(2)}(g) &\equiv T^{(2)} := T[:, 2, :] \\ \text{TP}^{(3)}(g) &\equiv T^{(3)} := T[:, 6, 7]\end{aligned}$$

**Listing 1.1.** Example SPARQL Query: Who of my friends knows a unicorn.

```
PREFIX :     <http://example.org/>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbr:  <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?f WHERE {
    :e1 foaf:knows ?f .          # TP(1)
    ?f  foaf:knows ?u .          # TP(2)
    ?u  rdf:type   dbr:Unicorn   # TP(3)
}
```

**Listing 1.2.** Example RDF Graph

```
@prefix :     <http://example.org/> .
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dbr:  <http://dbpedia.org/resource/> .

:e1 foaf:knows :e2 , :e3 .
:e2 foaf:knows :e3 , :e4 ;
    rdf:type   dbr:Unicorn .
:e3 foaf:knows :e2, :e4 .
:e4 rdf:type   dbr:Unicorn .
```

**Table 1.** Mapping from RDF terms to IDs.

| term | $id$(term) |
|---|---|
| :e1 | 1 |
| foaf:knows | 2 |
| :e2 | 3 |
| :e3 | 4 |
| :e4 | 5 |
| rdf:type | 6 |
| dbr:Unicorn | 7 |
| *unbound* | 8 |

**Table 2.** Mapping from RDF terms $\langle$s, p, o$\rangle$ to tensor keys $\langle id(\text{s}), id(\text{p}), id(\text{o})\rangle$ IDs.

| s | p | o | $id$(s) | $id$(p) | $id$(o) |
|---|---|---|---|---|---|
| :e1 | foaf:knows | :e2 | 1 | 2 | 3 |
| :e1 | foaf:knows | :e3 | 1 | 2 | 4 |
| :e2 | foaf:knows | :e3 | 3 | 2 | 4 |
| :e2 | foaf:knows | :e4 | 3 | 2 | 5 |
| :e3 | foaf:knows | :e2 | 4 | 2 | 3 |
| :e3 | foaf:knows | :e4 | 4 | 2 | 5 |
| :e2 | rdf:type | dbr:Unicorn | 3 | 6 | 7 |
| :e4 | rdf:type | dbr:Unicorn | 5 | 6 | 7 |

The resulting solution mappings of the TPs are:

$$\begin{aligned}
\text{TP}^{(1)}(g) &= \{[\text{?f} \to \text{:e2}], [\text{?f} \to \text{:e3}]\} \\
\text{TP}^{(2)}(g) &= \{[\text{?f} \to \text{:e1}, \text{?f} \to \text{:e2}], [\text{?f} \to \text{:e1}, \text{?u} \to \text{:e3}], [\text{?f} \to \text{:e2}, \text{?u} \to \text{:e3}], \\
&\qquad [\text{?f} \to \text{:e2}, \text{?u} \to \text{:e4}], [\text{?f} \to \text{:e3}, \text{?u} \to \text{:e2}], [\text{?f} \to \text{:e3}, \text{?u} \to \text{:e4}]\} \\
\text{TP}^{(3)}(g) &= \{[\text{?u} \to \text{:e2}], [\text{?u} \to \text{:e4}]\}
\end{aligned}$$

The corresponding slices $T^{(1)}$, $T^{(2)}$ and $T^{(3)}$ of $T$ are shown in Figure 2.

The joins between the TPs and the projection (`SELECT`) from Listing 1.1 are represented using Einstein Notations. The slices $T^{(1)}$, $T^{(2)}$ and $T^{(3)}$ are used as operands to the Einstein Notation, the variables of the corresponding TPs as subscripts and the variable(s) of the projection as subscript to the result $R$:

$$R_f \leftarrow T^{(1)}_f \times T^{(2)}_{f,u} \times T^{(3)}_u \tag{1}$$

The result $R$ is an order-1 tensor, which is calculated as $R[f \in \mathbb{I}_8] = \sum_{u \in \mathbb{I}_8} T^{(1)}[f] \cdot T^{(2)}[f,u] \cdot T^{(3)}[u]$, and results in $R = \langle \underset{1}{0}, \underset{2}{0}, \underset{3}{1}, \underset{4}{2}, \underset{5}{0}, \underset{6}{0}, \underset{7}{0}, \underset{8}{0} \rangle$. The vector encodes, that the result of SPARQL query contains once the solution mapping that maps ?f to the resource encoded by the ID 3 (:e2) and twice the solution mapping that maps ?f to the

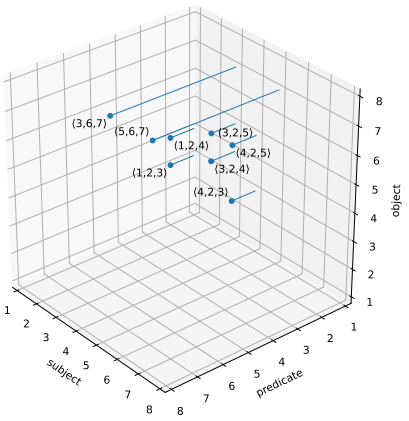


**Fig. 1.** 3D plot of the tensor $T$ that represents the RDF graph from Listing 1.2 with the term-to-ID mapping from Table 1. Every 1 is indicated by a point at the corresponding position.

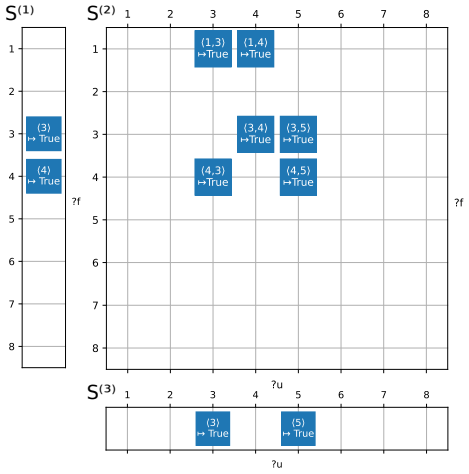


**Fig. 2.** Plots of slices $T^{(1)} = T[1, 2, :]$, $T^{(2)} := T[:, 2, :]$ and $T^{(3)} := T[:, 6, 7]$ of $T$ from Figure 1

resource encoded by ID 4 (:e3). Hence, the bag of solution mappings is:

$$\{[?f \rightarrow :e2], [?f \rightarrow :e3], [?f \rightarrow :e3]\}$$

## 3 RDF Tensor as Hypertrie

A hypertrie $h$ storing the tensor $T$ is shown in Figure 3.

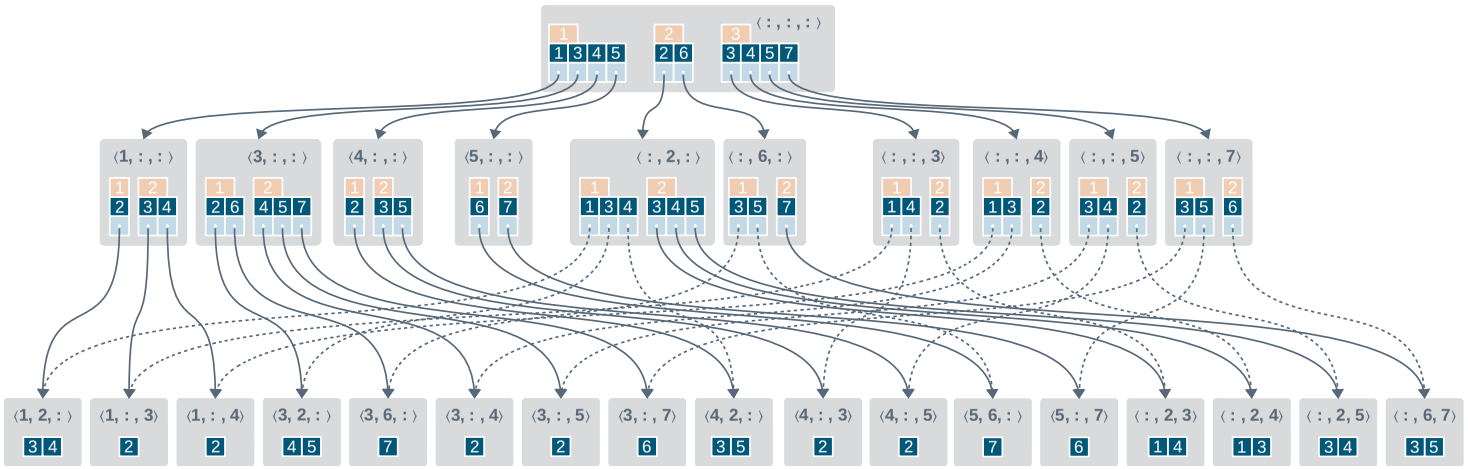


**Fig. 3.** The hypertrie $h$ that stores the tensor $T$ from Figure 1.

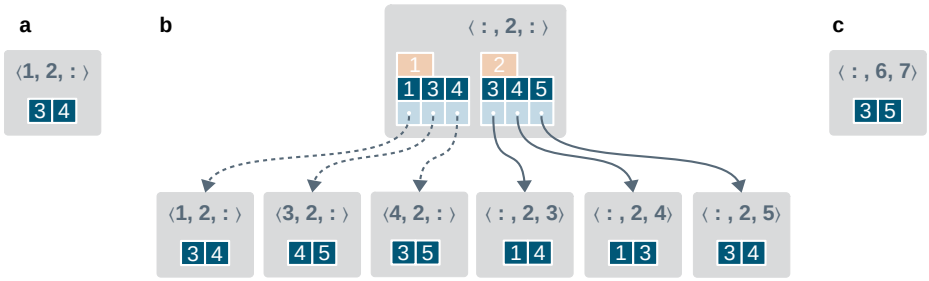


**Fig. 4.** Sub-hypertries a, b and c of $t$ that represent the slices $T^{(1)}$, $T^{(2)}$ and $T^{(3)}$ from Figure 2.

## 4 Slices On Hypertrie

The slices $T^{(1)}$, $T^{(2)}$ and $T^{(3)}$ are stored in sub-hypertries of $h$. The exact sub-hypertries are shown in Figure 4.

## 5 Einstein Summations on Hypertrie

An example of the execution of Equation (1) is given in Listing 1 as a call graph. In this example we use the tensor slicing notation, e.g. $h[1, 2, :]$, on the hypertrie $h$ to refer to the matching subhypertrie.

In the call graph the calculation the reduction factors for different labels is required only once in the call of `einsum_rek` in line 2 to decide between the labels $f$ and $u$ to be processed next. In all subsequent calls there is at most one label available for selection.

For `einsum_rek` in line 2 the reduction factors are calculated as follows:

$$
\begin{aligned}
m^{-}_{\langle h[1,2,:],h[:,2,:],h[:,6,7]\rangle,\langle\langle f\rangle,\langle f,u\rangle,\langle u\rangle\rangle}(f) &= \min(2,3) = 2\\
m^{-}_{\langle h[1,2,:],h[:,2,:],h[:,6,7]\rangle,\langle\langle f\rangle,\langle f,u\rangle,\langle u\rangle\rangle}(u) &= \min(3,2) = 2\\
m^{+}_{h[1,2,:],\langle f\rangle}(f) &= \max(2) = 2\\
m^{+}_{h[:,2,:],\langle f,u\rangle}(f) &= \max(3) = 3\\
m^{+}_{h[:,6,7],\langle u\rangle}(f) &= \max()\\
m^{+}_{h[1,2,:],\langle f\rangle}(u) &= \max()\\
m^{+}_{h[:,2,:],\langle f,u\rangle}(u) &= \max(3) = 3\\
m^{+}_{h[:,6,7],\langle u\rangle}(u) &= \max(2) = 2\\
\Phi_{\langle h[1,2,:],h[:,2,:],h[:,6,7]\rangle,\langle\langle f\rangle,\langle f,u\rangle,\langle u\rangle\rangle}(f) &= \frac{2}{2}\cdot\frac{2}{3}\cdot 1 = \frac{2}{3}\\
\Phi_{\langle h[1,2,:],h[:,2,:],h[:,6,7]\rangle,\langle\langle f\rangle,\langle f,u\rangle,\langle u\rangle\rangle}(u) &= 1\cdot\frac{2}{3}\cdot\frac{2}{2}\cdot\frac{2}{3}
\end{aligned}
$$

Since both labels have the same reduction factor and also both labels occur in the same amount of operands, we pick one randomly. In Listing 1, $f$ is picked.

**Listing 1:** Function call sequence and hierarchy for evaluating the Einstein summation in Equation (1). For better understanding, some local variables' values and changes to the result are given between the function calls.

```
1  einsum(⟨h[1,2,:], h[:,2,:], h[:,6,7]⟩, ⟨⟨f⟩, ⟨f,u⟩, ⟨u⟩⟩, ⟨f⟩)
2      einsum_rek(⟨h[1,2,:], h[:,2,:], h[:,6,7]⟩, ⟨⟨f⟩, ⟨f,u⟩, ⟨u⟩⟩, ⟨f⟩, ⟨8⟩,
         r = ⟨0,0,0,0,0,0,0,0,0⟩)
3          U = {f,u}
4          l = f
5          K = {3,4} ∩ {1,3,4} = {3,4}
6          κ = 3
7          einsum_rek(⟨1, h[3,2,:], h[:,6,7]⟩, ⟨⟨⟩, ⟨u⟩, ⟨u⟩⟩, ⟨f⟩, ⟨3⟩,
             r = ⟨0,0,0,0,0,0,0,0,0⟩)
8              U = {u}
9              l = f
10             K = {4,5} ∩ {3,5} = {5}
11             κ = 5
12             einsum_rek(⟨1,1,1⟩, ⟨⟨⟩, ⟨⟩, ⟨⟩⟩, ⟨f⟩, ⟨3⟩, r = ⟨0,0,0,0,0,0,0,0,0⟩)
13                 U = {}
14                 r[3]+← 1
15         κ = 4
16         einsum_rek(1, ⟨h[4,2,:], h[:,6,7]⟩, ⟨⟨⟩, ⟨u⟩, ⟨u⟩⟩, ⟨f⟩, ⟨4⟩,
             r = ⟨0,0,1,0,0,0,0,0,0⟩)
17             U = {u}
18             l = f
19             K = {3,5} ∩ {3,5} = {3,5}
20             κ = 3
21             einsum_rek(⟨1,1,1⟩, ⟨⟨⟩, ⟨⟩, ⟨⟩⟩, ⟨f⟩, ⟨4⟩, r = ⟨0,0,1,0,0,0,0,0,0⟩)
22                 U = {}
23                 r[4]+← 1
24             κ = 5
25             einsum_rek(⟨1,1,1⟩, ⟨⟨⟩, ⟨⟩, ⟨⟩⟩, ⟨f⟩, ⟨4⟩, r = ⟨0,0,1,1,0,0,0,0,0⟩)
26                 U = {}
27                 r[4]+← 1
28     return r = ⟨0,0,1,2,0,0,0,0,0⟩
```