Design for negative atoms

I shall start the explanation from the design in CastagnaPaper and then extend that to the more general case in FrishThesis. In page 61 of the CastagnaPaper he provide 2 representations for negative atoms as,

1. For open records

$$not(\{l_1 : T_1, l_2 : T_2, ..\}) = \{l_1? : not(T_1), ..\}|\{l_2? : not(T_2), ..\}$$

   (i.e union of records where l_1 is either `undefined` or $not(T_1)$ and l_2 is either `undefined` or $not(T_2)$).
2. For closed records

$$not(\{l_1 : T_1, l_2 : T_2\}) = \{l_1? : not(T_1), ..\}|\{l_2? : not(T_2), ..\}|\{l_1 : T_1, l_2 : T_2, +\}$$

   (I believe there is a typo in the last term in the paper)
   This is union of all records in which l_1 is of type $not(T_1)$ or `undefined`, records in which l_2 is of type $not(T_2)$ or `undefined` and **records in which l_1 is T_1 and l_2 is T_2 and there is at least another field defined.**

I believe 1 is sufficient for nBallerina as well (though we may choose to represent every thing using 2) and we can already represent this using our representation of mappings. So I'll limit the rest to extending 2 to match ballerina's requirements. Since we can already represent the first and second components of the union we shall focus only on the representation of the final component.

In the foot note 29 (in page 47) Castagna calls these records "Quasi-open records" defined as,

$$\{l_1 : T_1, \ldots, l_n : T_n, +\} = \{l_1 : T_1, \ldots, l_n : T_n, ..\} \setminus \{l_1 : ANY, \ldots, l_n : ANY\}$$

My intuition here is that in order to get the set of records where there must be "not-undefined" fields other than l_1 to l_n what we are doing is, take the set of records where where fields other than l_1 to l_n can be either undefined or any and subtract the set of records where fields other than l_1 to l_n is undefined.

Now let's extend this to our idea of open records where the rest must be of a given type T_{rest}. Let,

$$not(\{l_1 : T_1, l_2 : T_2, ..T_{rest}\}) = \{l_1? : not(T_1), ..\}|\{l_2? : not(T_2), ..\}|R^*$$

where R^* is the set of records where l_1 is T_1, l_2 is T_2 and **there is at least another field whose type is not T_{rest}.**

Based on the foot note 29 I think this R^* is (a less general version of) the R in FrishThesis page 177, given as.

$$R((t_l)_{l \in L}; t_0; E) = \{l_1 = t_{l_1}; \ldots; l_n = t_{l_n}; \_ = t_0\} \ \vee_{s \in E} \{l_1 = 1; \ldots; l_n = 1; \_ = \neg s\}$$

If my intuition for the "Quasi-open records" is correct what we need to do to get R^* is take the set of records where l_1 is T_1 , l_2 is T_2 and rest is `any` and subtract from it set of all records where rest is T_{rest}. That is in terms of R we will have t_0 equal to `any` and $E = \{T_{rest}\}$.

Example

Now to take as example lets consider `not {| byte x; byte y; ...byte|}`. This is equivalent to,

```
{| not(byte) x|} | {|not(byte) y|} | ({| byte x; byte y; ...any |} - {
any x; any y; ...byte});
```

That is the union of records where `x` is not `byte` or undefined (I assume not to include undefined in ballerina) , `y` is not `byte` or undefined and records where `x` is `byte` and `y` is `byte` and there is at least one field who is not `byte`. (That is $R(t; any; byte)$)

Implementation details

In the previous section I limited the usage of R only to places where we can not represent what we need without R. But based on theorem 9.5 in FrishThesis we should be able to represent all records as a union of R. (How to do this is not entirely clear to me at the moment). Therefore in the actual implementation I think it makes more sense to unify the representation of all records under R (similar to how we have unified the representation of open and closed records under open records).