Implementation of range matches for structural variants

With the integration of

- `variant_type`
- `start`
- `end`
- `svlen`

into GA4GH/schemas, there is now incentive to use comparable syntax for Beacon queries, and has been implemented in the upcoming version of the BeaconAlleleRequest.

Introduction

While, based on reference genomecoordinates, base specific genome variations can be described by a combination of a single genome position together with reference and alternate base(s), structural genome variations are described through the coordinates of the variants' start and end positions, together with the variants' types. These positions may not always consist of a single base but represent an approximate position of high probability (e.g. due to mapping problems, methods w/o complete genome coverage such as WES, arrays …). In the VCF specification, these *Variant Position Intervals* are denoted through the "CIPOS, CIEND" intervals for start and end of a structural variant, respecyively.

Importantly, in contrast to specific & recurring SNVs (think BRAF V600E), structural variants (think TP53 deletions in cancer) tend to be selected due to their functional effect (think all deletions rendering TP53 non-functional) rather than at events of a base-specific size. Therefore, *Query Ranges* are needed allow a "fuzzy" matching of all variants with a specific type, but varying genomic extent.

To summarize, the concept of intervals for variant start and end can be applied to two main scenarios:

1. When the start and end positions of a given genomic variant cannot be established exactly (i.e. involving DNA repeats; incomplete coverage/resolution of the used method)
2. In the case of a range query which tries to identify variants with possible differences in their exact start and end positions, e.g. all deletions somehow affecting the CDR of a gene

The following discussion will address point 2, and how it can be addressed through use of *Query Ranges* for start (and optional end) positions of variants of interest.

Point 1, *Variant Position Intervals*, is in the GA4GH schema e.g. addressed through the `cipos`, `ciend` attributes, derived from their representation in the VCF INFO field. *This is not part of this specific discussion.*

Genome Range Matches

Matching genome ranges when querying for variants requires the logical definition of how overlaps of the *Query Ranges* with variants are handled.

In this proposal,

- "start" is a list representation of the current single integer `BeaconAlleleRequest.start` attribute.
- "start" has a length of 1-2
- "end" is a list for querying the end position of a (structural) variant
- "end" has a length of 0-2
- These names are only for demonstration purposes, and in the schema would be just `start,end`

The positions in start and end each bracket the intervals for $\mathbf{V_{POS}}$ and $\mathbf{V_{END}}$ on the reference genome.

The server will return all matches for variant **V**, where

- $\mathbf{V_{POS}}$ is in the interval [start[0], start[1]]
- $\mathbf{V_{END}}$ is in the interval [end[0], end[1]]

Legend for graphics

- `?????` : bases corresponding to $\mathbf{V_{POS}}$ interval of the example query
- `?????` : bases corresponding to $\mathbf{V_{END}}$ interval of the example query
- `-----` : bases corresponding to the genome variant being matched

Basic query: DEL;Start=[15,21];End=[45,53]

```
Beacon request:

   start: [15,21]

   end: [45,53]

   variant_type: "DEL"
```

```
+++++++++++++++++++------------------------++++++++++++++++++++=>

              ???????                      ?????????
```

Interpretation: Direct submission of ranges in which $\mathbf{V_{POS}}$ and $\mathbf{V_{END}}$ have to occur.

Matched variants:

```
++++++++++++++++---------------------------++++++++++++++++++++=>

+++++++++++++++------------------------------------++++++++++++=>

++++++++++++++++++++-----------------------++++++++++++++++++++=>
```

Logical ranges and their interpolation into query ranges

In practical applications, frequently specific "*match types*" are being used. Understanding of these logical types and their application is helpful in understanding range match queries and their interpolation into specific query attributes.

For continuous ranges, a match can for example have one of the following "*matchtype*" options. This is just for demonstarting the use in an interface, which then will send start and end parameters to the server:

- *any*
    - Any overlap >= 1 between the range of the query and the variant
- *complete*
    - The variant has to be at least completely covered by the query range
- *exact*
    - Start and End of both query and variant are base identical
- *leftopen*
    - While the start of the variant can have any position, the end has to match the query end
- *rightopen*
    - While the end of the variant can have any position, the start has to match the query start

Query Ranges can be used to accommodate for all "*matchtype*" options, including such not specified above (e.g. combination one side exact, other side any etc.).

The following examples will demonstrate the interpolation of logical *matchtypes* into start and end ranges. No "matchtype" parameter is being sent to the server; the interpolation has to be provided through the interface logic.

Generally, "matchtype" represents an abstract - but frequently used - logical construct which is only applied here to demonstrate different query types.

Match Examples (against a genome of length 1000):

Basic query: DEL;Start=20;End=45

```
Beacon request:

   start: [20,20]

   end: [45,45]

   variant_type: “DEL”

+++++++++++++++++++?----------------------?++++++++++++++++++++=>
```

Interpretation: We are looking for a deletion DEL from position 20 and 45. Since no matchtype is provided, the interpretation is as in the “exact” match.

Basic query: DEL;Start=20;END=45,any

```
Beacon request:

   start: [0,45]

   end: [20,1000]

   variant_type: “DEL”

+++++++++++++++++++------------------------++++++++++++++++++++=>

????????????????????????????????????????????

                   ?????????????????????????????????????????????=>
```

Matched variants:

```
++++++++++++-------------++++++++++++++++++++++++++++++++++++++=>

------------------------------------------------------------++=>

+++++++++++++++++++-++++++++++++++++++++++++++++++++++++++++++++=>

++++++++++++++++++++++++++++++++++++++++--++++++++++++++++++++++=>
```

Interpretation: Any type of overlap with the basic DEL 20-45 is returned. This condition becomes true for any deletion DEL that starts between position 0 and the end of the region (45), and ends anywhere between the start of the region (20) and the end of the genome (1000). These conditions are emulated through providing

- start[0,45]: matches from genome start (0) to last base of the region (45)
- end[20,1000]: matches from first base of region to the end of the genome

Basic query: DEL;Start=20;END=45,complete

```
Beacon request:

   start: [0,20]

   end: [45,1000]

   variant_type: “DEL”
```

```
++++++++++++++++++------------------------+++++++++++++++++++=>

????????????????????                     ?????????????????????=>
```

Matched variants:

```
----------------------------------------------------------++=>

++++++++++++++++++--------------------------++++++++++++++++=>
```

Interpretation: We allow deletions to start *before or including* the start position and end *including or after* the end position. Note that the deletion needs to be completely within the defined window [20, 45], and therefore needs to have a minimal length of 25.

Basic query: DEL;Start=20;END=45,exact

```
Beacon request:

   start: [20,20]

   end: [45,45]

   variant_type: “DEL”
```

```
++++++++++++++++++------------------------+++++++++++++++++++=>

                  ?                      ?
```

Matched variants (only one possibility):

```
++++++++++++++++++------------------------+++++++++++++++++++=>
```

Interpretation: We want an **exact** match, i.e. the service should only return data that falls exactly into the range [20, 45].

This would be the default interpretation (however, for structural variants the main query interists will be for the “any” or “complete” types)..

Basic query: DEL;Start=20;END=45,rightopen

```
Beacon request:

   start: [20,20]

   end: [45,1000]

   variant_type: “DEL”
```

```
+++++++++++++++++++------------------------++++++++++++++++++++=>

                   ?                       ?????????????????????=>
```

Matched variants:

```
+++++++++++++++++++--------------------------------------------=>

+++++++++++++++++++------------------------++++++++++++++++++++=>
```

Interpretation: We require deletions to start *at* the start position and cover the whole range until the end position. They can end *at or after* the end position.

If no END is specified, this will lead to a simple one sided match:

Basic query: DEL;Start=20,rightopen

```
Beacon request:

   start: [20,20]

   end: [20,1000]

   variant_type: “DEL”
```

```
+++++++++++++++++++------------------------++++++++++++++++++++=>

                   ?????????????????????????????????????????????=>
```

Matched variants:

```
+++++++++++++++++++--------------------------------------------=>

+++++++++++++++++++------+++++++++++++++++++++++++++++++++++++++=>
```

Basic query: DEL;Start=20;END=45,leftopen

```
Beacon request:

   start: [0,20]

   end: [45,45]

   variant_type: “DEL”
```

```
+++++++++++++++++++-------------------------+++++++++++++++++++++=>

????????????????????                        ?
```

Matched variants:

```
+++++++++++++++++++-------------------------+++++++++++++++++++++=>

--------------------------------------------+++++++++++++++++++++=>
```

Interpretation: We allow deletions to start *at or before* the start position. They end *at* the end position.

Interpretation of CIPOS,CIEND parameters provided in the variant set:

The interpolation of match types into query-side start,end parameters is independent of `cipos,ciend` (in VCF: CIPOS,CIEND) values provided as part of the variant annotations. The recommendation here is to use the interpolated ranges for the start and end positions, and match them against the variant side intervals.