Scoring Document

Last Edited - June 8, 2011

For this review, we will be using

- SuffSmall(A,x) to denote the conditional that for attribute A and numerical x, $(A \leq x)$.
- pair(R,S) to denote a paired information relationship

Domain and Range

We have a growing number of Minerals with a finite number of attributes that are tied to these minerals. It is our concern to design a (search/scoring algorithm) and (database) which with as little deviance as possible addresses the Attributes as our domain for retrieving minerals.

Floating Point	***Integer***	***Enumerated***	***Color***
aspect_ratio_zy	twovx	cleavage_shape	pleochroism_col$_1$
aspect_ratio_xy	cleavage_num	twin$_i$_type	pleochroism_col$_2$
index_of_refraction$_\alpha$	cleavage_angle	true_optics	pleochroism_col$_3$
index_of_refraction$_\beta$	blasticity	opaque	anomalous_interference_col
index_of_refraction$_\gamma$	dispersion_strength	rock_type	
birefringence_num			

Attribute Catagories

There are several different Attribute Catagories. The Attribute Catagories are divided by overall topic and are made openly apparent in the UI.

1. Optical Character
2. Grain Shape
3. Index of Refraction
4. Pleochroism
5. Cleavage
6. Twinning
7. Birefringence
8. Abbundance

These are sectioned off in the UI by topic, this does not imply a lack of relation between data retrieved within differing Attribute Catagories, It is only a model of organizing for user ease.

Attribute Types

The greater concern for our database operations is the Attribute Types. Every unique type must have a slightly different search algorithm. Listed are the currently identified.

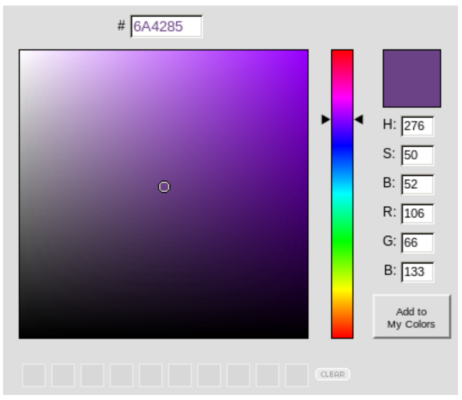
1. Color Attribute
2. Enumerated Attribute
3. Prune Attribute
4. Range Attribute

Color Attribute

We have two major themes for searching colors in the database. After bringing these two methods to David Hirsch's attention, he asked us to focus on other Attribute Types more so the documentation on this Attribute Catagories will be brief. For this category we are currently assuming that the colors are stored and addressed as HTML Color codes. We also note for full disclosure that in the mineral structure for every color there exists a variability number that we will call the CVstored in hexadecimal form.

Method 1

The first option for color identification is to allow a user to select from a standard color picker like the one `"http://www.colorpicker.com/"`. The Color-Picker layout is not important as long as it is continuous gradient for this model.



It is the algorithm's job determine if the user's input #XXYYZZ color code is in each minerals #QQRRSS range as defined by the area in #QQRRSS - #CVCVCVx # QQRRSS+#CVCVCV. The computational cost of this operation is rather heavy, and the user could easily choose a color that is outside of all of these values because they are provided to him, however it is more likely that the user is happy with their choice.

Method 2

The second proposed approach to this problem is to create images displaying the greatest of overlapping domains in color areas of the minerals. For this I will use an example:

given Mineral_1 with color code #LLMMNN and CV_1

given Mineral_2 with color code #QQRRSS and CV_2

If the area of

{#LLMMNN ± $\#CV_1CV_1CV_1$} overlaps sufficiently with

{#QQRRSS ± $\#CV_2CV_2CV_2$}

then they will be represented in the same color blot, and both minerals will be scored equally. This method forces the user to select an *active* color region (if they are to choose a color) and allows categorical computations to happen prior to search cutting run time dramatically.

Enumerated Attribute

The enumerated types can easily be used for pruning in the general case, however there are certain combinations of attributes that will cause **False Negative** in user categorization. The enumerated search variables are as listed:

1. true_optics
2. cleavage_shape
3. twin_i_type

These are relatively static, but contain several logical conditions in searching.

true_optics

Bellow is listed situations that result in incorrect guesses with respect to true_optics.

- The true_optics attribute has one of three options : isotropic, uniaxial, biaxial.
- If SuffSmall(birefringence_num,.005) and (true_optics == isotropic) then (true_optics = uniaxial or biaxial) are marked **False Negative**
- If SuffSmall(birefringence_num,.005) and (true_optics == uniaxial) then (true_optics = isotropic) is marked **False Negative**
- If SuffSmall(twovx,10°) and (true_optics == uniaxial) then true_optics = biaxial is marked **False Negative**
- if SuffSmall(twovx,10°) and (true_optics == biaxial) then true_optics = uniaxial is marked **False Negative**

cleavage_shape

cleavage_shape is a parameter that is mutually search-able with cleavage_num. Only one may be searched by a user. If they choose cleavage_shape it is assumed that they are more confident in their decision.

- cubic
- octahedral
- rhobohedral

twin_i_type

I have nothing to say here. I do not know if there are any unique issues with respect to assessing this enumerated type. David Hirsch should be questioned on this.

Prune Attribute

The majority of the Prune Attributes are found in Abbundance. These types are the last evaluated, and are used to scale active scores before displaying to the user. Ideally this grants common minerals, greater scores and more rare minerals lesser relative scores. There are a few ways to handle this procedure, and each has its merits.

he simple approach, and default approach. The only Prune Attribute that is search-able is rock_type. This is also the only search-able value in the Abbundance Category. This allows users to put in the type of rock they are inspecting to more accurately weight their returned values. The algorithm is generalized so if users don't choose rock_type the weights are drawn from (world_occurrence $\in$ Abbundance). [world_occurrence is the % of the earth that represents a given mineral].

A scenario that easily raises an issue: The user is identifying a rare mineral like Fluorite which has an world_occurrence of 2%, whose query also returns Quartz at 75%. If the scores provided for Fluorite and world_occurrence were scaled linearly, as percentages are expected to, it is likely that Quartz would win out, even if Fluorite were a better match, unless Quartz was a significantly lesser match. *Proposed solutions to this are :*

1. **scatter plot representation**
2. nonlinear scale - (perhaps logarithmic)
3. ability to disable score pruning
4. Secondary Sorting

The **scatter plot representation** removes the responsibility of pruning from the programmer. It would display as an x-axis the rock_type frequency ratings wrt returned minerals, and a y-axis of the scores they received. This allows the user to clearly identify where their mineral falls in it's worldly distribution. It's failing point would be when too many rare minerals are scored highly, the user would never receive high frequency minerals.

Alternatives for Prune Attribute

We have not enough data to test this method out, however it is possible that using logarithmic scale to the mineral's Prune Attributes would mellow out high scores enough to make the presented issue less relevant.

Disabling score pruning is by far the simplest means to implement, this lets the user decide what is more important. It also takes the responsibility out of the programmer for coming up with something clever. However, it is fair to argue that this should be a feature of the UI regardless.

The theme for secondary sorting is to identify highest scores first, separate sorted results into sections, then sort given sections using the Prune Attributes.

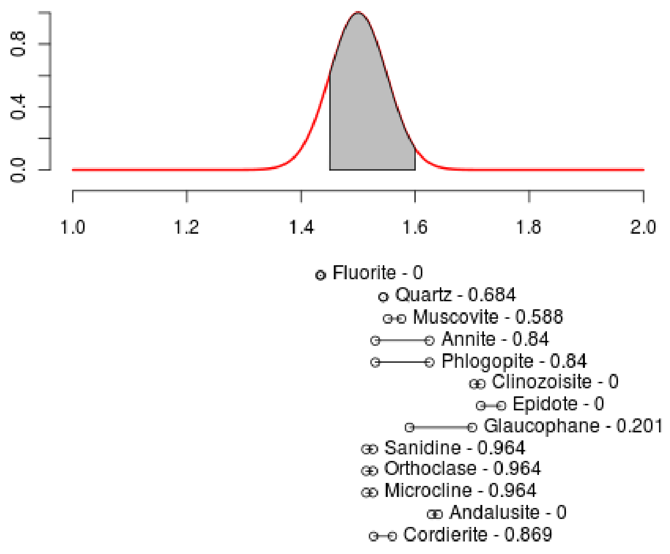
Range Attribute

The focus of our algorithm addresses Range Attributes. Range Attributes are individually searched with a 3:2 ratio. The user inputs their best guess, then provides a upper and lower bounds to their guess. This allows for much of the approximation to be the user's responsibility.

	Fluor	Quart	Musco	Annit	Phlog	Clino	Epido	Glauc	Sanid	Ortho	Micro	Andal	Cordi
Min	1.433	1.544	1.552	1.53	1.53	1.703	1.715	1.59	1.514	1.514	1.514	1.629	1.527
Max	1.435	1.544	1.576	1.625	1.625	1.715	1.751	1.7	1.526	1.526	1.526	1.64	1.56

*

The table above denotes Mineral's respective ranges for index_of_refraction$_\alpha$.



The Image above helps us represent a possible equation for scoring characteristics more specifically with respect to index_of_refraction$_\alpha$. The user has provided us with:

$\mu =$ 1.5 denoting mean or best guess

$\alpha =$ 1.45 denoting lower bound on guess

$\beta =$ 1.6 denoting upper bound on guess

The distribution curve is representative of the normal distribution as expressed by:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

In this instance we choose (could easily be altered or made into piecewise)

$$\sigma = min(|\mu - \alpha|, |\mu - \beta|)$$

The area defined by the shaded region is representative of the range addressed by the user, and the height is representative of how heavy the weighted score aught to be for a given mineral, whose index_of_refraction$_\alpha$ range is represented by unique lines below the graph.

Although it may be originally intuitive to use an integral equation to handle weighted scoring, there are several cases where one mineral's min and max of a specific attribute spans the entire domain. This can quickly throw off scores, so it is much more beneficial to just use the maximal value along the curve between the mineral's min and max.

Range Attribute Scoring

- Every attribute a has what we have called it's characteristic number $\phi(a)$. This is a number that denotes how representative of minerals an attribute is.
- M_i denotes the selected mineral.
- Every mineral has $R(M_i, a)$ which is the min and max value tied to the specific pair(attribute,mineral).
- $S(M_i)$ denotes a specific mineral's accumulated score

The score awarded to minerals by Range Attributes is:

$$S(M_i) = \sum_{a \in Attributes} \left[\phi(a) \cdot max \left(\frac{1}{f(\mu)} \cdot \frac{1}{\sqrt{2\pi {\sigma_a}^2}} \cdot e^{-\frac{(x-\mu_a)^2}{2{\sigma_a}^2}},\ x \in \text{range } (R(M_i, a) \cap \alpha(a)..\beta(a)) \right) \right]$$

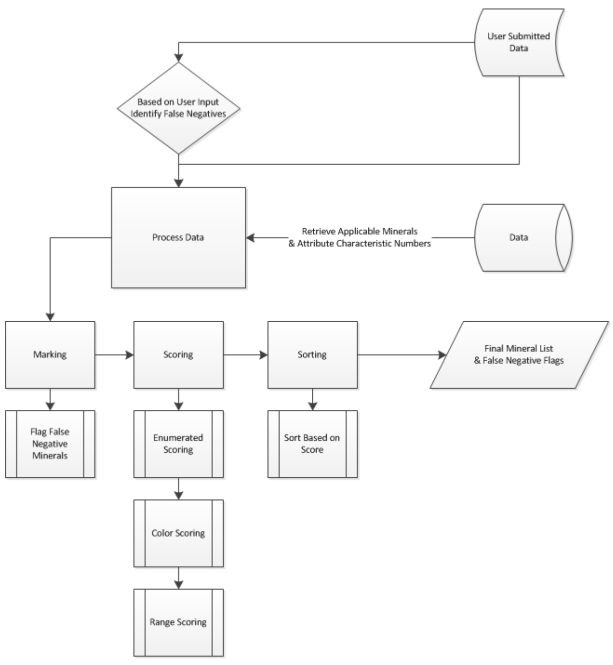
Because there are at most **5** unique comparable values per pair(mineral,attribute), there exist trivial ways to cut time on this computation.

To facilitate an easier UI, it would be advantageous to allow the user to just place in their best guess, then use predetermined $\alpha(a)$ and $\beta(a)$.

Flow Chart

The returned array has every relevent mineral, their total scores, and their catagorical scores.

- All Minerals start with a score of zero
- **False Negative & Marking** is done using the rules from the Enumerated Attributes
- **Enumerated Attribute Scoring** is completed by adding the $\phi(a)$ of the Enumerated Attribute a to the Mineral's Score
- **Range Attribute Scoring** is done using values generated by the equation in the Range Attribute Scoring Section


User Submitted Data
Based on User Input Identify False Negatives
Process Data
Retrieve Applicable Minerals & Attribute Characteristic Numbers
Data
Marking
Scoring
Sorting
Final Mineral List & False Negative Flags
Flag False Negative Minerals
Enumerated Scoring
Sort Based on Score
Color Scoring
Range Scoring

If the user requests N minerals to be returned. Then the returned sorted array should contain N elements that are specifically not **False Negative**. In other words, assuming the lowest score of the N requested minerals is 9:

if mineral R has score 20

and mineral S has score 9

and mineral R is marked **False Negative**

then S counts towards the N requested minerals

and R does not count towards the requested N minerals

however R is returned in the final array, so users can dynamically switch between displaying **False Negative**

also any mineral Q with equivalent minimal scores ($score(Q) = score(S)$) should be returned as well, and displayed regardless of user specifications because they are computationally identical.