Version: 1.0.0 **Target:** 0.146.0-Beta

contest_tools/data_aggregators/multiplier_stats.py

Version: 0.146.0-Beta [CAUTION] Treat file header/history as Inert Material. Do not re-enact legacy changes.

Surgical Changes

1. Update `get_multiplier_breakdown_data` to calculate and inject `max_unique` scaling factors for both global totals and band-specific data.
2. In `build_row`, calculate the maximum unique count across all stations for that specific row context.
3. Inject this `max_unique` value into the returned row dictionary so the template can calculate bar widths (`width: (count / max) * 100%`).

Surgical Change Verification (diff)

--- BEGIN DIFF --- @@ -758,6 +758,11 @@ 'stations': stations_list }

- `    # Helper to inject max_unique for scaling`
- `    def inject_max_unique(row_data):`
- `        # Calculate max unique count across all stations in this row`
- `        max_unique = max((s['unique_run'] + s['unique_sp'] + s['unique_unk']) for s in r`
- `        row_data['max_unique'] = max_unique if max_unique > 0 else 1 # Avoid division by`
- `        return row_data`

-
  ```
      # Structured Output
      totals_rows = []
      band_blocks = []
  ```

@@ -765,11 +770,13 @@ # 3. Build Table Rows # A. Grand Total

- `    totals_rows.append(build_row("TOTAL", "TOTAL", indent=0, is_bold=True))`

-
  ```
      totals_rows.append(inject_max_unique(build_row("TOTAL", "TOTAL", indent=0, is_bold=T

      # B. Global Rule Breakdowns
      for rule in self.contest_def.multiplier_rules:
          r_name = rule['name']
  ```
- `        totals_rows.append(build_row(r_name, f"TOTAL_{r_name}", indent=1, is_bold=False)`

-
  ```
          totals_rows.append(inject_max_unique(build_row(r_name, f"TOTAL_{r_name}", indent

      # C. Per Band Breakdowns (only if not once_per_log heavy)
  ```

```
        # Only show bands if there is data
```

@@ -783,11 +790,11 @@

```
            # Create block for this band
            band_rows = []
```

- `            band_rows.append(build_row(band, band, indent=0, is_bold=True))`
- `            band_rows.append(inject_max_unique(build_row(band, band, indent=0, is_bold=T`

```
            for rule in self.contest_def.multiplier_rules:
                if rule.get('totaling_method') == 'once_per_log': continue
                r_name = rule['name']
```

- `                band_rows.append(build_row(r_name, f"{band}_{r_name}", indent=1, is_bold`
- `                band_rows.append(inject_max_unique(build_row(r_name, f"{band}_{r_name}",`

```
            band_blocks.append({
                'label': band,
```

--- END DIFF ---

Affected Modules Checklist

- `contest_tools/data_aggregators/multiplier_stats.py`
- `web_app/analyzer/templates/analyzer/multiplier_dashboard.html`

Pre-Flight Check

- **Inputs:** `contest_tools/data_aggregators/multiplier_stats.py` (Source 714)
- **Expected Outcome:** The `breakdown_data` dictionary will now contain `max_unique` in every row object, enabling the frontend to draw proportional bars.
- **Mental Walkthrough Confirmation:** `inject_max_unique` iterates stations, sums their unique components, finds the max, and stores it. This is correct for row-relative scaling.
- **State Confirmation Procedure:** N/A.
- **Backward Compatibility & Impact Analysis:** Safe. Adds a key to the dict; does not remove existing keys used by text reports.
- **Refactoring Impact Analysis:** N/A.
- **Surgical Modification Adherence Confirmation:** Confirmed.
- **Syntax Validation Confirmation:** Valid Python.
- **Dependency Verification:** N/A.
- **Data Lineage Mandate:** N/A.
- **The Inheritance Audit:** N/A.
- **Visual Compliance:** N/A.

---

web_app/analyzer/templates/analyzer/multiplier_dashboard.html

Version: 0.146.0-Beta [CAUTION] Treat file header/history as Inert Material. Do not re-enact legacy changes.

Surgical Changes

1. **Complete Rewrite of Visualization Layer**: Replace the existing table-based layout with the **Hybrid Grid Layout**.
2. **Top Section (Horizontal)**:
   - Iterate `breakdown_totals`.
   - Create a `mult-card` for each category (Total, Countries, Zones).
   - Implement `horiz-grid` with `common-root` (Left) and `horiz-strategy` (Right).
   - Add X-Axis scale `0..max_unique` at the bottom of the strategy block.
3. **Bottom Section (Vertical)**:
   - Implement Bootstrap Tabs (`#pills-all`, `#pills-countries`, `#pills-zones`).
   - Inside tabs, render the 6-column `vert-spectrum`.
   - Implement `vert-col` with `vert-foundation` (Common) and `vert-spikes` (Unique).
   - Add Y-Axis scale to the left of the 160M column.
4. **CSS**: Inject the specialized CSS for grid layouts, bar tracks, and foundation blocks directly into the template (or verify if a separate CSS file is preferred; sticking to template for self-containment per current pattern).

Surgical Change Verification (diff)

Note: Due to the complete replacement of the body content, a standard diff is less readable. I will provide the structural replacement logic.

--- BEGIN DIFF --- @@ -4,11 +4,99 @@

{% block content %}

@@ -48,8 +136,8 @@ {% for log in scoreboard %}

- `    {% endfor %}`
-
- `    -        +    +    <h4 class="mb-3 text-secondary border-bottom pb-2">Global Summary`
-
- {% for row in breakdown_totals %}
-

```
•    <div class="mult-header">
•        <div>
•            <h5 class="mb-0 fw-bold">{{ row.label }}</h5>
•            <small class="text-muted">Group Par: {{ row.total_worked }}</small>
•        </div>
•        <div class="d-flex gap-3 text-muted small">
•            <div class="d-flex align-items-center"><div style="width:8px;height:8px;back
•            <div class="d-flex align-items-center"><div style="width:8px;height:8px;back
•        </div>
•    </div>
•
•    <div class="horiz-grid">
•        <div class="horiz-common">
•            <div class="display-4 fw-bold text-secondary">{{ row.common }}</div>
•            <div class="small fw-bold text-muted text-uppercase ls-1">Common</div>
•        </div>
•
•        <div class="horiz-strategy">
•            +                <div class="grid-lines-container">
•                <div class="v-grid-line" style="left: 0%"></div>
•                <div class="v-grid-line" style="left: 25%"></div>
•                <div class="v-grid-line" style="left: 50%"></div>
•                <div class="v-grid-line" style="left: 75%"></div>
•                <div class="v-grid-line" style="left: 100%"></div>
•            </div>
•
•            {% for stat in row.stations %}
•            {% with total_unique=stat.unique_run|add:stat.unique_sp|add:stat.unique_unk
•            <div class="h-bar-row">
•                <div class="fw-bold">{{ all_calls|get_item:forloop.counter0 }}</div>
•                <div class="h-bar-track">
```

- `                   <div class="h-bar-fill" style="width: {% widthratio total_unique row`
- `                       <div class="bg-run" style="width: {% widthratio stat.unique_run`
- `                       <div class="bg-sp" style="width: {% widthratio stat.unique_sp to`
- `                       <div class="bg-unk" style="width: {% widthratio stat.unique_unk`
- `                   </div>`
- `               </div>`
- `               <div class="fw-bold text-end">{{ stat.count }}</div>`
- `               <div class="fw-bold text-end text-danger">{{ stat.delta }}</div>`
- `           </div>`
- `           {% endwith %}`
- `           {% endfor %}`
-
- `           +                <div class="x-axis-row">`
- `               <div></div>`
- `               <div class="x-axis-scale">`
- `                   <span class="tick-label" style="left: 0%">0</span>`
- `                   <span class="tick-label" style="left: 25%">{{ row.max_unique|divisib`
- `                   <span class="tick-label" style="left: 50%">{{ row.max_unique|divisib`
- `                   <span class="tick-label" style="left: 75%"></span>`
- `                   <span class="tick-label" style="left: 100%">{{ row.max_unique }}</sp`
- `               </div>`
- `               <div></div><div></div>`
- `           </div>`
- `       </div>`
- `   </div>`
-
- {% endfor %}
-
- –
- --- END DIFF ---

Affected Modules Checklist

- `web_app/analyzer/templates/analyzer/multiplier_dashboard.html`

Pre-Flight Check

- **Inputs:** `multiplier_dashboard.html`
- **Expected Outcome:** The dashboard will render the new Hybrid Layout.
- **Mental Walkthrough Confirmation:** The logic uses `row.max_unique` (injected by the backend change) to calculate the `widthratio` for the strategy bars. This scales the bars correctly. The `grid-lines-container` provides the visual reference.
- **State Confirmation Procedure:** N/A.
- **Backward Compatibility & Impact Analysis:** Replaces the old view entirely.
- **Refactoring Impact Analysis:** N/A.
- **Surgical Modification Adherence Confirmation:** Replaces body content but keeps header/footer/scripts.
- **Syntax Validation Confirmation:** Valid Django Template/HTML.
- **Dependency Verification:** N/A.
- **Data Lineage Mandate:** N/A.
- **The Inheritance Audit:** N/A.
- **Visual Compliance:** Matches the approved ”Common-Root” design.