```
################################################################################
# This file is part of polap.
#
# polap is free software: you can redistribute it and/or modify it under the
# terms of the GNU General Public License as published by the Free Software
# Foundation, either version 3 of the License, or (at your option) any later
# version.
#
# polap is distributed in the hope that it will be useful, but WITHOUT ANY
# WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR
# A PARTICULAR PURPOSE. See the GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License along with
# polap. If not, see <https://www.gnu.org/licenses/>.
################################################################################

################################################################################
# Ensure that the current script is sourced only once
source "$script_dir/run−polap−function−include.sh"
_POLAP_INCLUDE_=$(_polap_include "${BASH_SOURCE[0]}")
[[ -n "${!_POLAP_INCLUDE_}" ]] && return 0
declare "$_POLAP_INCLUDE_=1"
#
################################################################################

################################################################################
# Selects contigs for an organelle-genome assembly.
#
# 1. We could select mitochondrial- or plastid-derived contigs using a contig annotation table.
# 2. We determine the range of sequencing depths for those candidate contigs: mean +/- sd \* 3.
# 3. For a given gfa of a genome assembly graph, subset the graph for selecting graph elements in the range.
# 4. Determine connected components in the subset.
# 5. Choose connected components with candidate edges.
#
# We need to read GFA files to manipulate.
# We need to determine connected components.
################################################################################
function _run_polap_select-contigs-by() {
        # Enable debugging if DEBUG is set
        [ "$DEBUG" -eq 1 ] && set -x
        _polap_log_function "Function start: $(echo $FUNCNAME | sed s/_run_polap_//)"

        # Set verbosity level: stderr if verbose >= 2, otherwise discard output
        local _polap_output_dest="/dev/null"
        [ "${_arg_verbose}" -ge "${_polap_var_function_verbose}" ] && _polap_output_dest="/dev/stderr"

        # Grouped file path declarations
        source "$script_dir/polap−variables−mtcontig.sh" # ’.’ means ’source’

        # Print help message if requested
        help_message=$(
                cat <<HEREDOC
# Selects contigs using three features using total lengths of contigs.
#
# Use the total length in the cumulative distribution of contig lengths
# to select the lower and upper bounds of contig lengths.
#
# To identify seed contigs of mitochondrial origin,
# a whole-genome assembly is evaluated for three criteria:
# 1) the presence of mitochondrial or plastid genes,
# 2) the number of read coverage, and
# 3) the connectivity of contigs in the genome assembly graph.
#
# 1. We could select mitochondrial- or plastid-derived contigs using a contig annotation table.
# 2. We determine the range of sequencing depths for those candidate contigs: mean +/- sd \* 3.
#   2.1 Construct the cumulative distribution of contig lengths.
#   2.2 Given L1=3Mb, determine the lower bound of the contig length.
#   2.3 Given L2=300 kb, determine the upper bound of the contig length.
# 3. For a given gfa of a genome assembly graph, subset the graph for selecting graph elements in the range.
# 4. Determine connected components in the subset.
# 5. Choose connected components with candidate edges.
#
# Arguments:
#   -i $INUM: source Flye (usually whole-genome) assembly number
#   -j $JNUM: destination Flye organelle assembly number
#   --select-contig: 1 ~ 5
#
# Inputs:
#   ${_polap_var_assembly_graph_final_gfa}
#   ${_polap_var_annotation_table}
#
# Outputs:
#   $MTCONTIGNAME
#   "${_polap_var_mtcontig_annotated}"
#
# See:
#   run-polap-select-contigs-by-table-1.R for the description of --select-contig option
Example: $(basename $0) ${_arg_menu[0]} [-i|--inum <arg>] [-j|--jnum <arg>] [--select-contig <number>]
Example: $(basename $0) ${_arg_menu[0]} -o PRJNA914763 -i 0 -j 5 --select-contig 5
HEREDOC
        )

        # Display help message
        [[ ${_arg_menu[1]} == "help" ]] && _polap_echo0 "${help_message}" && exit $EXIT_SUCCESS

        # Display the content of output files
        if [[ "${_arg_menu[1]}" == "view" ]]; then

                case "${_arg_select_contig}" in
                1 | 3)
                        check_file_existence "${_polap_var_mtcontig_annotated}"
                        _polap_log0_cat "${_polap_var_mtcontig_annotated}"
                        check_file_existence "${_polap_var_mtcontig_stats}"
                        _polap_log0_cat "${_polap_var_mtcontig_stats}"
                        ;;
                2 | 4 | 5)
                        check_file_existence "${_polap_var_mtcontig_mixfit}"
                        _polap_log0_cat "${_polap_var_mtcontig_mixfit}"
                        check_file_existence "${_polap_var_mtcontig_annotated}"
                        _polap_log0_cat "${_polap_var_mtcontig_annotated}"
                        check_file_existence "${_polap_var_mtcontig_stats}"
                        _polap_log0_cat "${_polap_var_mtcontig_stats}"
                        check_file_existence "${_polap_var_mtcontig_mixfit}"
                        _polap_log0_cat "${_polap_var_mtcontig_mixfit}"
                        ;;
                *)
                        echo "Invalid input!"
                        ;;
                esac

                _polap_log2 "Function end: $(echo $FUNCNAME | sed s/_run_polap_//)"
                # Disable debugging if previously enabled
                [ "$DEBUG" -eq 1 ] && set +x
                exit $EXIT_SUCCESS
        fi

        _polap_log0 "selecting seed contigs using $(echo $FUNCNAME | sed s/_run_polap_//) ${INUM} −> ${JNUM} with type ${_arg_select_contig}"

        # Check for required files
        check_file_existence "${_polap_var_assembly_graph_final_gfa}"
        check_file_existence "${_polap_var_annotation_table}"
        _polap_log0 " input1: ${_polap_var_assembly_graph_final_gfa}"
        _polap_log0 " input2: ${_polap_var_annotation_table}"

        # Clean and create working directory
        _polap_log1 " delete and create dir:${_polap_var_mtcontigs}"
        _polap_log3_cmd rm -rf "${_polap_var_mtcontigs}"
        _polap_log3_cmd mkdir -p "${_polap_var_mtcontigs}"

        # Step 1: Determine the depth range using the cumulative length distribution.
        # Step 1: Select contigs based on genes
        _polap_log1 " select−contig type: ${_arg_select_contig}"
        _polap_log1 "run−polap−select−contigs−by−table−1.R"
        _polap_log2_file " input1: ${_polap_var_annotation_table}"
        _polap_log2_file " output−base1: ${_polap_var_mtcontig_base}"
        case "${_arg_select_contig}" in
        1 | 3)
                "$WDIR"/run-polap-select-contigs-by-table-1.R \
                        -t "${_polap_var_annotation_table}" \
                        -o "${_polap_var_mtcontig_base}" \
                        -c -d 10 \
                        2>"$_polap_output_dest"
                ;;
        2 | 4)
                "$WDIR"/run-polap-select-contigs-by-table-1.R \
                        -t "${_polap_var_annotation_table}" \
                        -o "${_polap_var_mtcontig_base}" \
                        -c -d 10 \
                        -r \
                        2>"$_polap_output_dest"
                ;;
        5)
                "$WDIR"/run-polap-select-contigs-by-table-1.R \
                        -t "${_polap_var_annotation_table}" \
                        -o "${_polap_var_mtcontig_base}" \
                        -c -d 10 \
                        -r \
                        -s \
                        2>"$_polap_output_dest"
                ;;
        *)
                echo "Invalid input!"
                ;;
        esac

        _polap_log2_file " output1: ${_polap_var_mtcontig_stats}"
        _polap_log2_file " output2: ${_polap_var_mtcontig_annotated}"

        case "${_arg_select_contig}" in
        1 | 2)
                # Save the first column (contig names) to the output file
                if [ -s "${_polap_var_mtcontig_annotated}" ]; then
                        cut -f1 "${_polap_var_mtcontig_annotated}" |
                                sort | uniq >"${MTCONTIGNAME}"
                        _polap_log1_file "output: ${MTCONTIGNAME}"
                else
                        >"${MTCONTIGNAME}"
                        _polap_log1_file "output: empty ${MTCONTIGNAME}"
                fi
                _polap_log2 "Function end: $(echo $FUNCNAME | sed s/_run_polap_//)"
                [ "$DEBUG" -eq 1 ] && set +x
                return
                ;;
        3 | 4 | 5)
                if [ ! -s "${_polap_var_mtcontig_annotated}" ]; then
                        >"${MTCONTIGNAME}"
                        _polap_log1_file "output: empty ${MTCONTIGNAME}"
                        _polap_log2 "Function end: $(echo $FUNCNAME | sed s/_run_polap_//)"
                        [ "$DEBUG" -eq 1 ] && set +x
                        return
                fi
                ;;
        *)
                echo "Invalid input!"
```

```
			;;
		esac

		# Handle case with single starting contig
		local mtcontig_count=$(wc -l <"${_polap_var_mtcontig_annotated}")
		if [ "${mtcontig_count}" -eq 1 ]; then
			cut -f1 "${_polap_var_mtcontig_annotated}" >"${MTCONTIGNAME}"
			_polap_log2_log "single starting contig"
			_polap_log1_file "output: ${MTCONTIGNAME}"
			_polap_log2 "Function end: $(echo $FUNCNAME | sed s/_run_polap_//)"
			[ "$DEBUG" -eq 1 ] && set +x
			return
		fi

		# Extract sequences and filter GFA data
		_polap_log2 "creating GFA without sequence data"
		gfatools view -S "${_polap_var_assembly_graph_final_gfa}" \
			>"${_polap_var_gfa_all}" \
			2>"$_polap_output_dest"
		_polap_log2_file "${_polap_var_gfa_all}"

		_polap_log2 "extracting sequence part of GFA"
		gfatools view -S "${_polap_var_assembly_graph_final_gfa}" \
			2>"$_polap_output_dest" |
			grep "^S" >"${_polap_var_gfa_seq_part}"
		_polap_log2_file "${_polap_var_gfa_seq_part}"

		# Filter edges in GFA using depths.
		_polap_log2 "filtering GFA sequence part using depth range"
		"$WDIR"/run-polap-select-contigs-by-depth-length-2-gfa-filter.R \
			"${_polap_var_gfa_seq_part}" \
			"${_polap_var_mtcontig_stats}" \
			"${_polap_var_gfa_seq_filtered}" \
			"${_polap_var_gfa_seq_filtered_range}" \
			2>"$_polap_output_dest"

		_polap_log2_file "${_polap_var_gfa_seq_filtered}"

		# Recreate GFA based on filtered edge sequences.
		_polap_log2 "subsetting GFA using the depth-filtered GFA sequence part"
		cut -f1 "${_polap_var_gfa_seq_filtered}" >"${_polap_var_gfa_seq_filtered_edge}"
		gfatools view -S \
			-l @"${_polap_var_gfa_seq_filtered_edge}" \
			"${_polap_var_assembly_graph_final_gfa}" 2>/dev/null \
			>"${_polap_var_gfa_filtered}"

		_polap_log2_file "${_polap_var_gfa_filtered}"

		# Prepare links for finding connected components.
		grep "^L" "${_polap_var_gfa_filtered}" | cut -f2,4 >"${_polap_var_gfa_links}"
		_polap_log2_file "${_polap_var_gfa_links}"
```

```

        # Run R script to analyze GFA links
        _polap_log2 "preparing for finding connected components"
        "$WDIR"/run-polap-select-contigs-3-gfa-links.R \
                "${_polap_var_mtcontig_annotated}" \
                "${_polap_var_gfa_links}" \
                "${_polap_var_gfa_links_number}" \
                "${_polap_var_gfa_links_order}" \
                "${_polap_var_gfa_links_contig}" \
                "${_polap_var_gfa_links_contig_na}" \
                2>"$_polap_output_dest"

        _polap_log2_file "${_polap_var_gfa_links_number}"
        _polap_log2_file "${_polap_var_gfa_links_order}"
        _polap_log2_file "${_polap_var_gfa_links_contig}"
        _polap_log2_file "${_polap_var_gfa_links_contig_na}"

        # Find connected components using Python script
        _polap_log2 "finding connected components by the depth-filtered contigs"
        python "$WDIR"/run-polap-select-contigs-4-find-connected-components.py \
                "${_polap_var_gfa_links_number}" \
                "${_polap_var_gfa_links_contig}" \
                "${_polap_var_gfa_links_seed}" \
                2>"$_polap_output_dest"

        _polap_log2_file "${_polap_var_gfa_links_seed}"

        # Choose final mitochondrial contigs
        _polap_log2 "converting the depth-filtered contigs in edge with numbers"
        "$WDIR"/run-polap-select-contigs-5-gfa-mtcontig.R \
                "${_polap_var_gfa_links_seed}" \
                "${_polap_var_gfa_links_order}" \
                "${_polap_var_gfa_links_mtcontig}" \
                2>"$_polap_output_dest"

        _polap_log2_file "${_polap_var_gfa_links_mtcontig}"

        _polap_log2 "concatenating the depth-filtered edges and NA edges?"
        cat "${_polap_var_gfa_links_mtcontig}" "${_polap_var_gfa_links_contig_na}" |
                sort | uniq >"${MTCONTIGNAME}"
        _polap_log1_file "output: ${MTCONTIGNAME}"

        "$WDIR"/run-polap-select-contigs-by-table-2.R \
                -t "${_polap_var_annotation_table}" \
                -m "${MTCONTIGNAME}" \
                -o "${_polap_var_mtcontig_base}" \
                2>"$_polap_output_dest"

        # _polap_log2_file " output1: ${_polap_var_mtcontig_stats}"
        _polap_log2_file " output2: ${_polap_var_mtcontig_annotated}"

        _polap_log2 "Function end: $(echo $FUNCNAME | sed s/_run_polap_//)"
        # Disable debugging if previously enabled
        [ "$DEBUG" -eq 1 ] && set +x
}
```